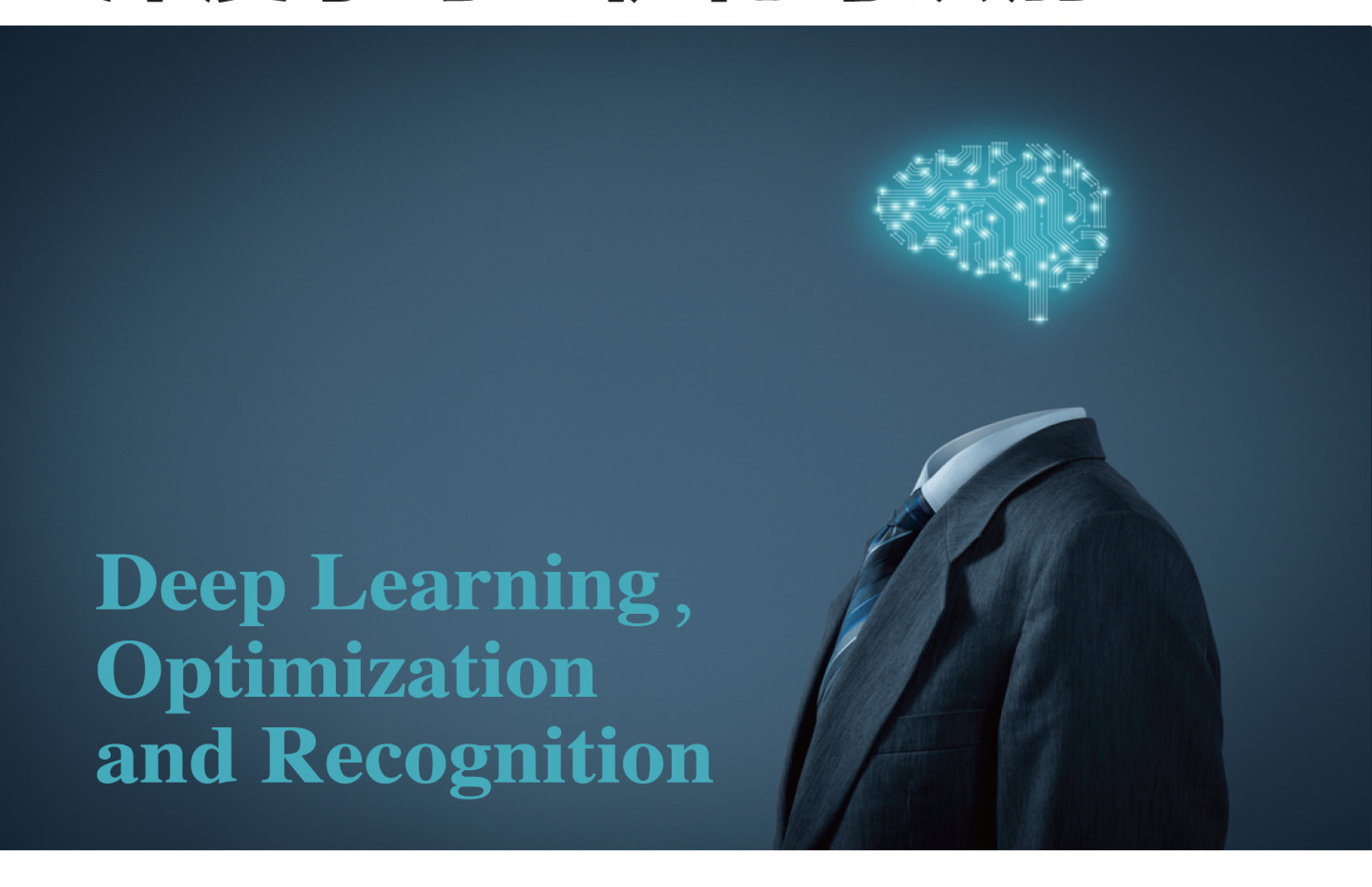
焦李成 赵 进 杨淑媛 刘 芳 著

深度学习、优化与识别



清华大学出版社

深度学习、优化与识别

Deep Learning, Optimization and Recognition

焦李成　赵　进　杨淑媛　刘　芳　著

清华大学出版社
北　京

内容简介

深度神经网络是近年来受到广泛关注的研究方向，它已成为人工智能 2.0 的主要组成部分。本书系统地论述了深度神经网络的基本理论、算法及应用。全书共 16 章，分为两个部分：第一部分(第 1 章～10 章)系统论述理论及算法，包括深度前馈神经网络、深度卷积神经网络、深度堆栈神经网络、深度递归神经网络、深度生成网络、深度融合网络等；第二部分(第 11 章～15 章)论述常用的深度学习平台，以及在高光谱图像、自然图像、SAR 与极化 SAR 影像等领域的应用；第 16 章为总结与展望，给出了深度学习发展的历史图、前沿方向及最新进展。每章都附有相关阅读材料及仿真代码，以便有兴趣的读者进一步钻研探索。

本书可为高等院校计算机科学、电子科学与技术、信息科学、控制科学与工程、人工智能等领域的研究人员提供参考，以及作为相关专业本科生及研究生教学参考书，同时可供深度学习及其应用感兴趣的研究人员和工程技术人员参考。

图书在版编目(CIP)数据

深度学习、优化与识别/焦李成等著. —北京：清华大学出版社，2017
ISBN 978-7-302-47367-1

Ⅰ. ①深…　Ⅱ. ①焦…　Ⅲ. ①人工神经网络—研究　Ⅳ. ①TP183

中国版本图书馆 CIP 数据核字(2017)第 097460 号

责任编辑：王　芳　薛　阳
封面设计：常雪影
责任校对：李建庄
责任印制：沈　露

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 装 者：三河市中晟雅豪印务有限公司
经　　销：全国新华书店
开　　本：186mm×240mm　　**印　　张**：25.75　　**字　　数**：624 千字
版　　次：2017 年 7 月第 1 版　　**印　　次**：2017 年 7 月第1 次印刷
印　　数：1～3000
定　　价：128.00 元

产品编号：075883-01

Preface.. 序 Preface..

从1308年加泰罗尼亚诗人、神学家雷蒙·卢尔(Ramon Llull)发表了有关用机械方法从一系列现象中创造新知识的论文开始,到1943年美国心理学家W.S. McCulloch和数学家W.Pitts提出MP模型及1950年A.Turing提出著名的图灵测试,再到1956年达特茅斯会议上人工智能的诞生,神经网络几经沉浮,走过了艰难曲折的历程;2006年从单隐层神经网络到深度神经网络模型,迎来了神经网络发展的又一高潮,深度学习及其应用受到了前所未有的重视与关注,世界迎来又一轮人工智能变革的高潮,从谷歌脑到中国脑科学计划,再到互联网+和中国人工智能2.0,人工智能及深度学习也首次写进了2017年全国人民代表大会第十五次会议国务院政府工作报告。深度学习是人工智能及机器学习的一个重要方向,在未来,它将会不断出现激动人心的理论进展和方法实践,深刻影响我们生活的方方面面。

随着研究的不断深入,深度学习已经超越了目前机器学习模型的神经科学观点,学习多层次组合的这一设计原则更加吸引人。从第一代的深度前馈神经网络开始,随之而来的就有如下三个问题:一是可用训练数据量远小于模型中的参数量,容易出现过(欠)拟合现象;二是随着层级的增加,模型的优化目标函数呈现高度非凸性,由于待优化参数所在的可行域中存在着大量的鞍点和局部极小值点,所以参数初始化策略影响着网络模型的稳定性和收敛性;三是基于误差的反向传播算法越靠近输出层变化越大,越靠近输入层变化越小,这对通过梯度下降方式来实现逐层参数更新会导致梯度弥散现象。为了解决第一个问题便提出了深度卷积神经网络和深度循环神经网络,其核心均是通过约减参数量间接提升数据量的方式降低过拟合现象的发生;针对第二个问题和第三个问题便引入了基于自编码器的逐层初始化策略,以期获取的初始化参数能够避免过早地陷入局部最优,同时弱化或克服梯度弥散现象,例如基于受限波尔兹曼机的深度置信网络。进一步,基于传统的机器学习算法来实

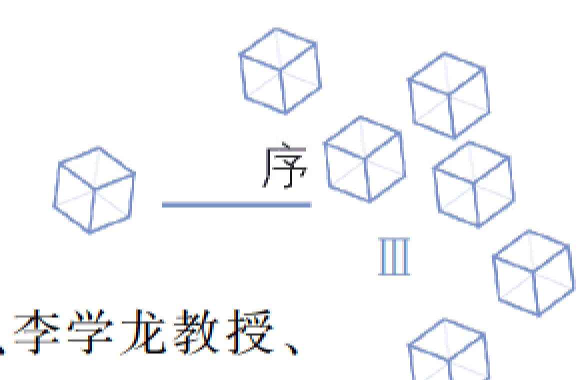


现参数初始化方向上涌现了如深度 PCA 网络、深度 ICA 网络、深度 SVM 网络、深度森林（随机森林多层级联）、深度极限学习机和深度 ADMM 网络等模型。同时与之类似的，通过更改非线性函数以换取模型“扭曲”能力的提升，产生了如深度小波网络、深度脊波网络和深度轮廓波网络等模型。根据其特性，我们称这些网络为深度融合网络。2014 年以来，大量的研究文献表明层级“深度”的不断增加，或导致性能显著提升（如深度残差网络、深度分形网络），抑或导致性能严重下降（本质上是参数量远大于训练数据量）。为了解决该问题，一方面通过多通路、并行化的网络设计来削弱“深度”对性能的依赖性，同时塔式结构、对称性等也被融入网络的设计过程中；另一方面，深度生成模型也悄然兴起，其核心是通过生成训练数据集的概率密度函数来实现数据的扩充，其代表便是生成式对抗网络和变分自编码器。值得注意的是，与传统的深度学习设计“单网络”不同，生成式对抗网络采用了“两个子网络”来实现非合作状态下的博弈，在最小最大值定理的保证下，理论上可以保证网络的收敛性。除了模型结构和优化策略改进外，应用问题背景也不再是经典的输入输出“单数据对”刻画，而是从状态到行动“整体性”刻画。众所周知，感知、认知和决策是衡量智能化的标准，充分发挥深度学习的感知能力和强化学习的决策能力，形成的深度强化学习已在众多应用问题上取得突破，如无人驾驶、计算机围棋程序和智能机器人等。在后深度学习时代，其核心在于生成数据、环境交互和领域迁移，对应着深度生成网络、深度强化学习和深度迁移学习将继续成为人工智能领域的研究热点。另外，根据数据的属性和操作的有效性，衍生的网络包括深度复数域神经网络（如深度复卷积神经网络）、深度二值神经网络和深度脉冲神经网络等。

我们依托智能感知与图像理解教育部重点实验室、智能感知与计算国际联合实验室及智能感知与计算国际联合研究中心于 2014 年成立了类脑计算与深度学习研究中心，致力于类脑计算与深度学习的基础与应用研究，搭建了多个深度学习应用平台，并在深度学习理论、应用及实现等方面取得了突破性的进展，本书即是我们在该领域研究工作的初步总结。

本书的完成离不开团队多位老师和研究生的支持与帮助，感谢团队中侯彪、刘静、公茂果、王爽、张向荣、吴建设、缑水平、尚荣华、刘波、刘若辰等教授以及马晶晶、马文萍、白静、朱虎明、田小林、张小华、曹向海等副教授对本工作的关心支持与辛勤付出。感谢王蓉芳博士、冯捷博士、张丹老师，以及唐旭、刘芳、谢雯、任博、魏野、王善峰、冯志玺等博士生在学术交流过程中无私的付出与生活上的关心。同时，特别感谢赵佳琦、刘旭、赵暐、朱浩、孙其功、任仲乐、李娟飞、张雅科、宋玮、张文华等博士生，以及马丽媛、杨争艳、张婷、李晰、孟繁荣、汶茂宁、侯瑶琪、孙莹莹、张佳琪、杨慧、王美玲等研究生在写作过程中无私付出的辛勤劳动与努力。感谢宋玮、张文华等博士生帮忙校勘时发现了许多笔误。

本书是我们团队在该领域工作的一个小结，也汇聚了西安电子科技大学智能感知与图像理解教育部重点实验室、智能感知与计算国际联合实验室及智能感知与计算国际联合研究中心的集体智慧。在本书出版之际，特别感谢邱关源先生及保铮院士三十多年来的悉心培养与教导，特别感谢徐宗本院士、张钹院士、李衍达院士、郭爱克院士、郑南宁院士、谭铁牛院士、马远良院士、包为民院士、郝跃院士、陈国良院士、韩崇昭教授，IEEE Fellows 管晓宏



教授、张青富教授、张军教授、姚新教授、刘德荣教授、金耀初教授、周志华教授、李学龙教授、吴枫教授、田捷教授、屈嵘教授、李军教授和张艳宁教授，以及马西奎教授、潘泉教授、高新波教授、石光明教授、李小平教授、陈莉教授、王磊教授等多年来的关怀、帮助与指导，感谢教育部创新团队和国家"111"创新引智基地的支持；同时，我们的工作也得到西安电子科技大学领导及国家"973"计划（2013CB329402）、国家自然科学基金（61573267，61472306，61671305，61573267，61473215，61571342，61572383，61501353，61502369，61271302，61272282，61202176）、重大专项计划（91438201，91438103）等科研任务的支持，特此感谢。同时特别感谢清华大学出版社的大力支持和帮助，感谢王芳老师和薛阳老师付出的辛勤劳动与努力。感谢书中所有被引用文献的作者。

20世纪90年代初我们出版了《神经网络系统理论》《神经网络计算》《神经网络的应用与实现》等系列专著，三十年来神经网络取得了长足的进展，本书的取材和安排完全是作者的偏好，由于水平有限，书中不妥之处恳请广大读者批评指正。

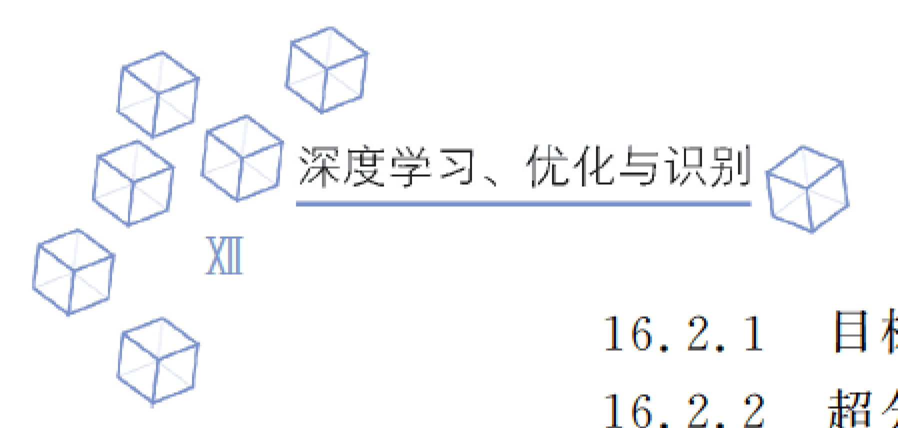
著　者

2017年3月

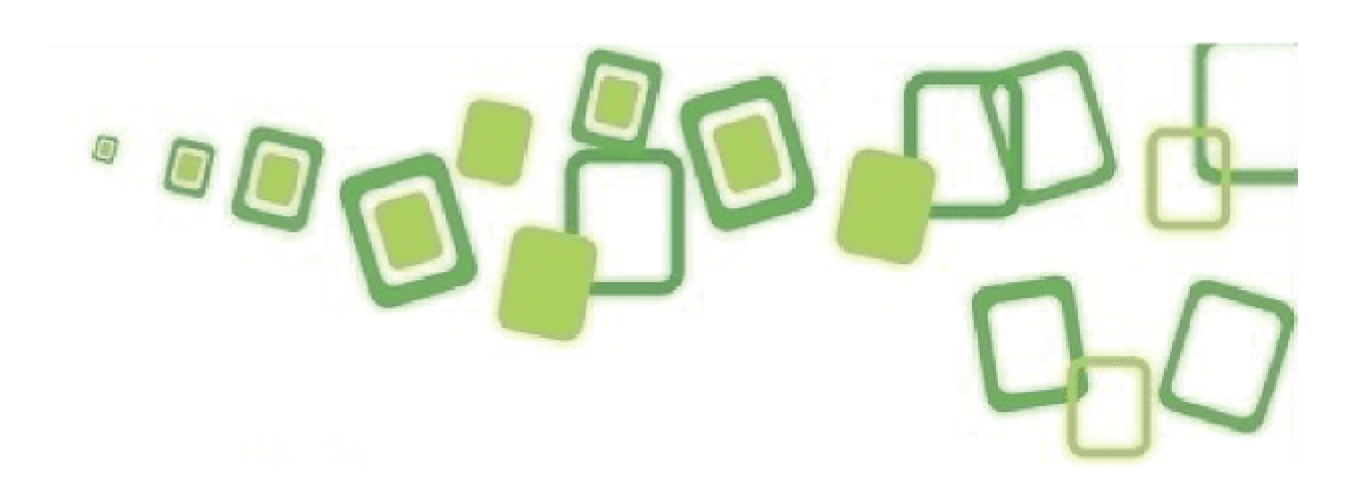
西安电子科技大学

目 录

Contents..

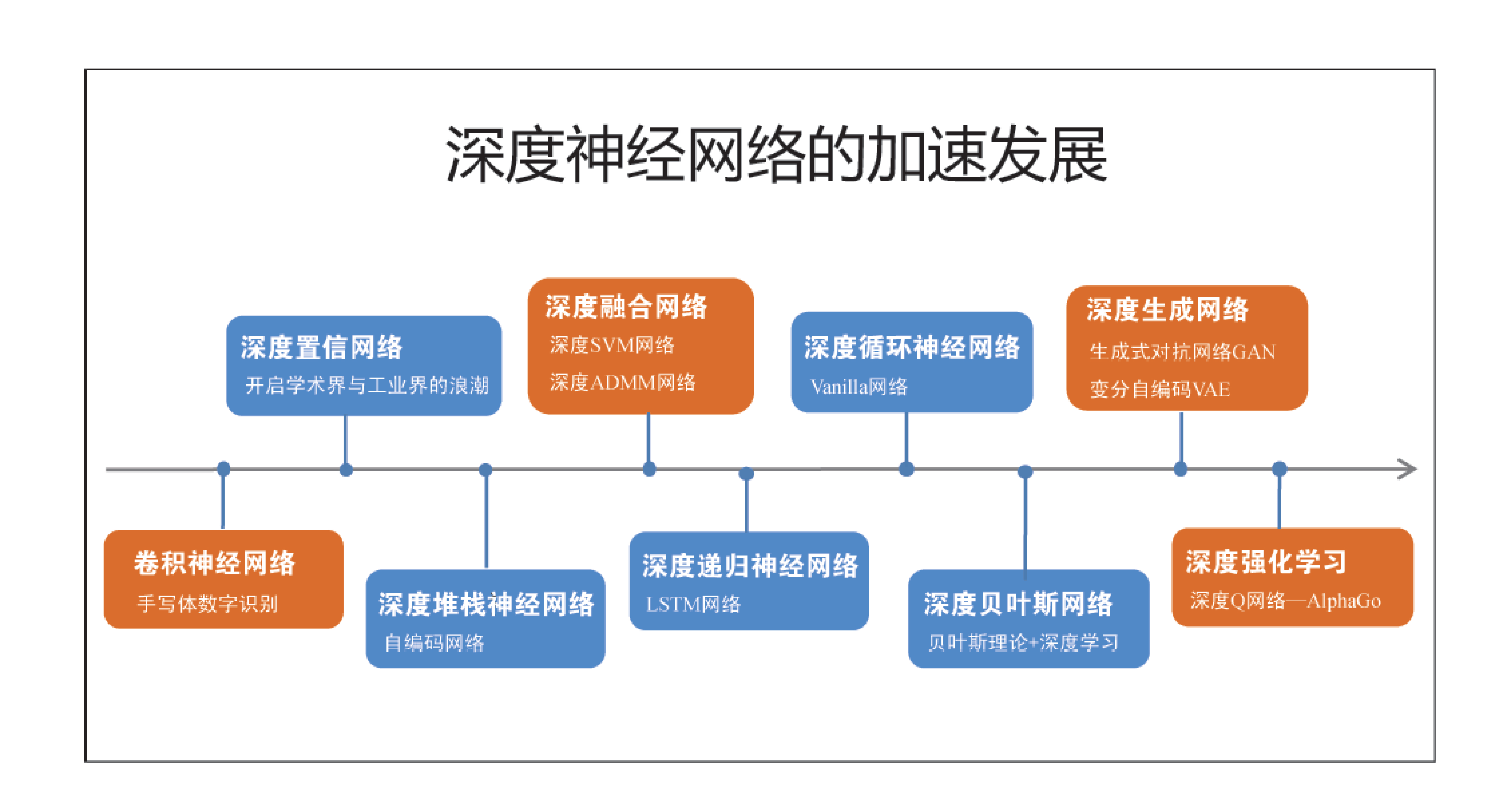


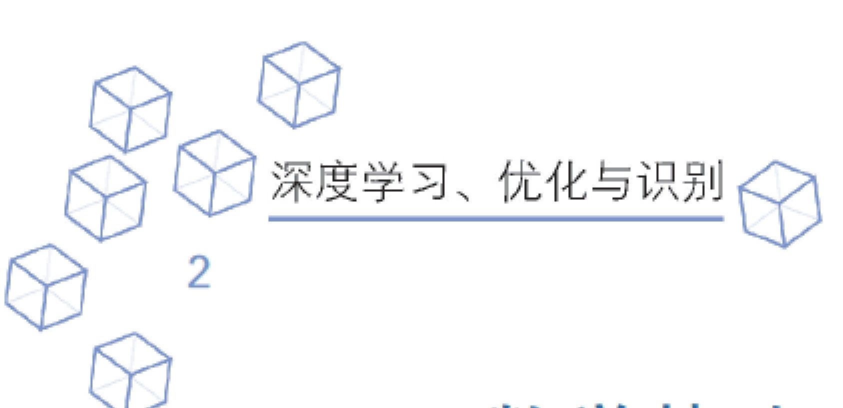
第 1 章



深度学习基础

CHAPTER 1





1.1 数学基础

1.1.1 矩阵论

在实数域上，大小为 $n\times m$ 矩阵的集合可以表示为：

$$\boldsymbol{M}(\mathbb{R}^{n\times m}) = \{\boldsymbol{A}:\boldsymbol{A}\in\mathbb{R}^{n\times m}\} \tag{1.1}$$

因此，$(\boldsymbol{M}(\mathbb{R}^{n\times m}),\mathbb{R})$可作为线性空间。为了刻画该空间中矩阵与矩阵之间的关系，则需要定义距离，如 $\forall\,\boldsymbol{A},\boldsymbol{B}\in\boldsymbol{M}(\mathbb{R}^{n\times m})$，它们的距离 distance$(\boldsymbol{A},\boldsymbol{B})$满足非负性，对称性和三角不等式性。通常，可以通过定义范数的形式来诱导距离，常用的范数有：$\forall\,\boldsymbol{A}\in\boldsymbol{M}(\mathbb{R}^{n\times m})$

$$\|\boldsymbol{A}\|_1 = \max\left\{\sum_{i=1}^{n}|\boldsymbol{A}_{i,1}|,\sum_{i=1}^{n}|\boldsymbol{A}_{i,2}|,\cdots,\sum_{i=1}^{n}|\boldsymbol{A}_{i,m}|\right\} \tag{1.2}$$

$$\|\boldsymbol{A}\|_2 = \boldsymbol{A}\text{ 的最大奇异值} \tag{1.3}$$

$$\|\boldsymbol{A}\|_F = \left(\sum_{i=1}^{n}\sum_{j=1}^{m}(\boldsymbol{A}_{i,j})^2\right)^{\frac{1}{2}} \tag{1.4}$$

$$\|\boldsymbol{A}\|_\infty = \max\left\{\sum_{j=1}^{m}|\boldsymbol{A}_{1,j}|,\sum_{j=1}^{m}|\boldsymbol{A}_{2,j}|,\cdots,\sum_{j=1}^{m}|\boldsymbol{A}_{n,j}|\right\} \tag{1.5}$$

$$\|\boldsymbol{A}\|_{1,2} = \sum_{i=1}^{n}\left(\sum_{j=1}^{m}(\boldsymbol{A}_{i,j})^2\right)^{\frac{1}{2}} \tag{1.6}$$

$$\|\boldsymbol{A}\|_{2,1} = \left(\sum_{i=1}^{n}\left(\sum_{j=1}^{m}|\boldsymbol{A}_{i,j}|\right)^2\right)^{\frac{1}{2}} \tag{1.7}$$

在实际的信号处理过程中，无论是构建损失项还是正则项，每一种范数都有其特定的物理意义，反映着数据的分布类型，或者蕴含着数据的先验特性。

由于在线性空间$(\boldsymbol{M}(\mathbb{R}^{n\times m}),\mathbb{R})$上通过范数诱导得到距离，所以便可以衡量其中任意两个矩阵的临近关系，即邻域特性，根据这种邻域特性，便可以将线性空间进行剖分，当然剖分的子空间的个数取决于邻域的半径。通常，称具有距离的线性空间为距离空间(赋范线性空间)。在机器学习中，该空间中的任意一个矩阵都可以视为是一种线性变换，当然非线性变换可以通过线性变换的逼近来得到，逼近的程度取决于范数的选取以及邻域半径的定义。

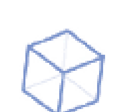
矩阵的导数的求解通常在机器学习中较为常用，如参数更新时所依赖的梯度的计算等。假设对于输入信号 $\boldsymbol{x}\in\mathbb{R}^m$ 与输出信号 $\boldsymbol{y}\in\mathbb{R}^n$ 之间存在着线性映射关系，即

$$f(\boldsymbol{x}) \triangleq \boldsymbol{A}\cdot\boldsymbol{x}+\boldsymbol{b}\approx\boldsymbol{y} \tag{1.8}$$

其中 $\boldsymbol{A}\in\mathbb{R}^{n\times m}$ 为“投影”矩阵，$\boldsymbol{b}\in\mathbb{R}^n$ 为偏置项。通常利用 L_2 范数来定义损失函数，即

$$\text{Loss}(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2}\|\boldsymbol{A}\cdot\boldsymbol{x}+\boldsymbol{b}-\boldsymbol{y}\|_2^2 \tag{1.9}$$

其中待学习的参数为$(\boldsymbol{A},\boldsymbol{b})$。由于损失项的形式是凸函数，所以损失项关于参数的偏导数



可以通过如下的公式求解：

$$\begin{cases}\dfrac{\partial \mathrm{Loss}}{\partial \boldsymbol{A}} = (\boldsymbol{A}\cdot\boldsymbol{x}+\boldsymbol{b}-\boldsymbol{y})\cdot\boldsymbol{x}^{\mathrm{T}} \\ \dfrac{\partial \mathrm{Loss}}{\partial \boldsymbol{b}} = (\boldsymbol{A}\cdot\boldsymbol{x}+\boldsymbol{b}-\boldsymbol{y})\end{cases} \tag{1.10}$$

进一步，对于参数的求解可以通过迭代更新来实现，即给定初始$(\boldsymbol{A}^{(0)},\boldsymbol{b}^{(0)})$及学习速率$\alpha$，通过下面的迭代公式进行更新：

$$\begin{cases}\boldsymbol{A}^{(k)} = \boldsymbol{A}^{(k-1)} - \alpha\cdot\dfrac{\partial \mathrm{Loss}}{\partial \boldsymbol{A}}\bigg|_{\boldsymbol{A}=\boldsymbol{A}^{(k-1)}} \\ \boldsymbol{b}^{(k)} = \boldsymbol{b}^{(k-1)} - \alpha\cdot\dfrac{\partial \mathrm{Loss}}{\partial \boldsymbol{b}}\bigg|_{\boldsymbol{b}=\boldsymbol{b}^{(k-1)}}\end{cases} \tag{1.11}$$

直至$(\boldsymbol{A}^{(k)},\boldsymbol{b}^{(k)})$收敛为止或者满足迭代终止条件。

另外，为了防止过拟合现象，通常会用富比尼斯范数约束“投影”矩阵作为正则项，即$\|\boldsymbol{A}\|_F^2$，它的导数为：

$$\frac{\partial\|\boldsymbol{A}\|_F^2}{\partial \boldsymbol{A}} = \frac{\partial \mathrm{tr}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})}{\partial \boldsymbol{A}} = 2\boldsymbol{A} \tag{1.12}$$

注意：这里的过拟合现象是指数据样本量相比参数量而言较多，导致训练得到的模型十分依赖于该数据集，使得该模型的测试性能或者预测性能比较差，即在另一数据集上的表现较差(需要说明的是这两个数据集的分布方式相同)。

矩阵的奇异值分解是指，对于任意一个矩阵$\boldsymbol{A}\in\mathbb{R}^{n\times m}$，都有如下的表达式：

$$\begin{cases}\boldsymbol{A} = \boldsymbol{U}\cdot\boldsymbol{\Sigma}\cdot\boldsymbol{V}^{\mathrm{T}} \\ \boldsymbol{U}^{\mathrm{T}}\cdot\boldsymbol{U} = \boldsymbol{I}_n \\ \boldsymbol{V}\cdot\boldsymbol{V}^{\mathrm{T}} = \boldsymbol{I}_m\end{cases} \tag{1.13}$$

其中，$\boldsymbol{\Sigma}$为对角矩阵，且$\boldsymbol{U}\in\mathbb{R}^{n\times n}$和$\boldsymbol{V}\in\mathbb{R}^{m\times m}$。在机器学习中，主成分分析便用到了矩阵的奇异值分解，通过奇异值的排序和信息利用率达到85%以上的准则确定主成分的个数；通常，主成分分析是一种线性的降维方法。使用矩阵的奇异值分解的核心是逼近的思想，可以通过调整对角矩阵$\boldsymbol{\Sigma}$中的值，实现对矩阵$\boldsymbol{A}$的刻画，这样既可以有效地对数据进行泛化，又可以达到降维、进而实现减少计算量的目的。

1.1.2　概率论

在机器学习的领域中，经常使用后验概率来实现执果索因的目的，常用的公式表述为：

$$\begin{cases}P(X\mid Y) = \dfrac{P(Y\mid X)P(X)}{\sum\limits_X P(Y\mid X)P(X)} \\ P(Y) = \sum\limits_X P(Y\mid X)P(X)\end{cases} \tag{1.14}$$

公式中，称$P(X|Y)$为随机事件Y发生的前提下，随机事件X发生的概率，也称为后验概率，$P(X)$为先验项或先验概率，$P(Y|X)$为似然项，$P(Y)$为随机变量Y的先验概率或边缘

概率，也称为标准化常量。进一步，公式 $P(Y)$ 可以根据完备空间的有限剖分及可数可加性得到执因索果的乘法公式。

最大似然估计针对模型已定、参数未知，提供了一种给定数据来评估模型参数的方法。假设数据集 $\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N$ 为独立同分布的采样，f 为已知的模型(如服从高斯分布，拉普拉斯分布等)，θ 为模型的参数。

根据独立同分布的假设：

$$P(\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N \mid \theta) = \prod_{i=1}^{N} P(\boldsymbol{x}_i \mid \theta) \tag{1.15}$$

其似然函数的定义为：

$$L(\theta \mid \boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N) = \prod_{i=1}^{N} P(\boldsymbol{x}_i \mid \theta) \tag{1.16}$$

参数 θ 的最大似然估计是通过最大化似然函数，使得求出的 θ 值与实际观察中的训练样本最相符，即

$$\max_{\theta} L(\theta \mid \boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N) \tag{1.17}$$

实际应用中，常利用最大化平均对数似然，即

$$\theta^* = \arg\max_{\theta\in\Theta} \frac{\ln(L(\theta \mid \boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_N))}{N} = \frac{\sum_{i=1}^{N}\ln(P(\boldsymbol{x}_i \mid \theta))}{N} \tag{1.18}$$

需要注意的是，最大似然估计仅是参数估计的方法之一，通过若干次独立同分布的实验，观察其结果，利用结果推算出参数的大概值。

对于分类问题，也可以利用最大似然估计来优化，考虑到计算问题，我们经常会使用最小化负对数似然损失函数，即 $\{\boldsymbol{x}_i,y_i\}_{i=1}^{N}$，其中 $\boldsymbol{x}_i\in\mathbb{R}^n$ 为输入，输出目标为 $y_i\in\{1,2,\cdots,C\}$；学习的模型为 $y=f(\boldsymbol{x},\theta)$，由于目标为离散的类别，所以通过计算输出每个类的条件概率来界定损失函数，即

$$\begin{cases} f_c(\boldsymbol{x},\theta) = \mathrm{P}(y=c \mid \boldsymbol{x},\theta), c = 1,2,\cdots,C \\ \sum_{c=1}^{C} f_c(\boldsymbol{x},\theta) = 1 \\ f_c(\boldsymbol{x},\theta) \in [0,1] \end{cases} \tag{1.19}$$

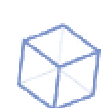
得到的负对数似然函数为：

$$l(y,f(\boldsymbol{x},\theta)) = -\sum_{c=1}^{C} y_c \log f_c(\boldsymbol{x},\theta) \tag{1.20}$$

进一步，最终的目标函数为：

$$\max_{\theta} L(\theta) = \frac{\sum_{i=1}^{N} l(y_i,f(\boldsymbol{x}_i,\theta))}{N} \tag{1.21}$$

通常该方法也被称为交叉熵损失函数。



1.1.3　优化分析

交替迭代乘子算法(ADMM)并不是一个很新的算法，它适用于求解分布式凸优化问题(其提出早于大规模分布式计算系统和大规模优化问题)，通过分解协调将大的全局问题分解为多个较小、容易求解的局部子问题，并通过协调子问题的解而得到大的全局问题的解。若优化问题可以表示为：

$$\begin{aligned}&\min_{\boldsymbol{x},\boldsymbol{z}} f(\boldsymbol{x})+g(\boldsymbol{z})\\&\text{s.t. } \boldsymbol{A}\cdot\boldsymbol{x}+\boldsymbol{B}\cdot\boldsymbol{z}=\boldsymbol{c}\end{aligned}\tag{1.22}$$

其中，$\boldsymbol{x}\in\mathbb{R}^s$，$\boldsymbol{z}\in\mathbb{R}^n$，$\boldsymbol{A}\in\mathbb{R}^{p\times s}$，$\boldsymbol{B}\in\mathbb{R}^{p\times n}$，$\boldsymbol{c}\in\mathbb{R}^p$，且 f,g 分别可以将 $\boldsymbol{x},\boldsymbol{z}$ 映射至$\mathbb{R}$。可以从 ADMM 的名字看出，通过引入新变量，然后交叉换方向来交替优化，即引入 $\boldsymbol{y}\in\mathbb{R}^p$，求解如下的目标函数：

$$\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} L_\rho(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})=f(\boldsymbol{x})+g(\boldsymbol{z})+\boldsymbol{y}^{\mathrm{T}}\cdot(\boldsymbol{A}\cdot\boldsymbol{x}+\boldsymbol{B}\cdot\boldsymbol{z}-\boldsymbol{c})+\frac{\rho}{2}\|\boldsymbol{A}\cdot\boldsymbol{x}+\boldsymbol{B}\cdot\boldsymbol{z}-\boldsymbol{c}\|_F^2\tag{1.23}$$

优化过程具有可分解性，通过如下的迭代公式更新求解：

$$\begin{cases}\boldsymbol{x}^{(k+1)}=\arg\min\limits_{x} L_\rho(\boldsymbol{x},\boldsymbol{y}^{(k)},\boldsymbol{z}^{(k)})\\\boldsymbol{z}^{(k+1)}=\arg\min\limits_{x} L_\rho(\boldsymbol{x}^{(k+1)},\boldsymbol{y}^{(k)},\boldsymbol{z})\\\boldsymbol{y}^{(k+1)}=\boldsymbol{y}^{(k)}+\rho\cdot(\boldsymbol{A}\cdot\boldsymbol{x}^{(k+1)}+\boldsymbol{B}\cdot\boldsymbol{z}^{(k+1)}-\boldsymbol{c})\end{cases}\tag{1.24}$$

注意，名称中的乘子算法指一种使用增广拉格朗日函数的(带有二次惩罚项)对偶上升方法，而交替迭代指的是 $\boldsymbol{x},\boldsymbol{z}$ 交替更新。那么如何保证 ADMM 算法的收敛性？需要具有如下两个假设条件：

(1) 函数 f,g 具有闭的、良态的、凸函数的性质；

(2) 拉格朗日函数 $L_\rho(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$在 $\rho=0$ 时有鞍点。

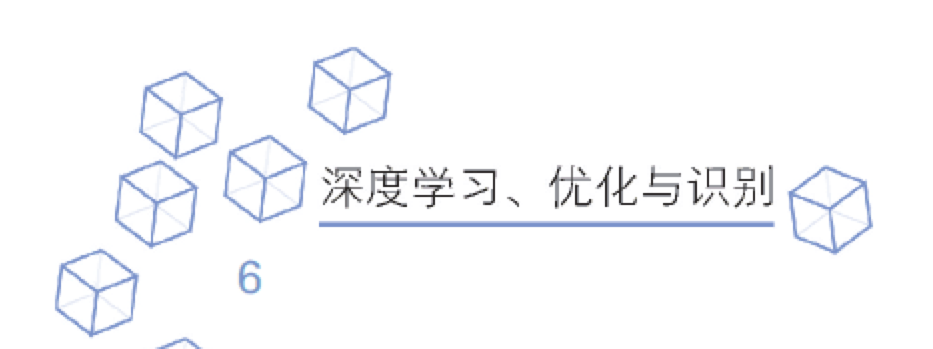
需要注意的是，在高精度要求下，ADMM 的收敛性很慢；但在中等精度要求下其速度可以接受。关于对偶问题，探索带有不等式约束的极值问题求解如下：

$$\begin{aligned}&\min_{\omega} f(\boldsymbol{\omega})\\&\text{s.t. } g_i(\boldsymbol{\omega})\leqslant 0,\quad i=1,2,\cdots,k\\&\quad\;\; h_i(\boldsymbol{\omega})=0,\quad i=1,2,\cdots,l\end{aligned}\tag{1.25}$$

对应的一般化拉格朗日公式为：

$$\min_{\omega,\alpha,\beta} L(\boldsymbol{\omega},\alpha,\beta)=f(\omega)+\sum_{i=1}^{k}\alpha_i\cdot g_i(\boldsymbol{\omega})+\sum_{i=1}^{l}\beta_i\cdot h_i(\boldsymbol{\omega})\tag{1.26}$$

这里 α_i 和 β_i 都是拉格朗日乘子。但是如果按照这个公式求解，会出现问题，因为求解的是最小值，而这里的 $g_i(\boldsymbol{\omega})$已经不为零了，可以将 α_i 调整为非常大的正值，来使得最后的结果为负无穷。为了排除这种情况，定义下面的函数：



$$\theta_{\text{Primal}}(\boldsymbol{\omega}) = \max_{\substack{\alpha:\alpha_i \geqslant 0 \\ \beta}} L(\boldsymbol{\omega}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{1.27}$$

这里之所以使用最大值，是因为对于 $g_i(\boldsymbol{\omega})>0$ 或者 $h_i(\boldsymbol{\omega})\neq 0$，总可以调整 α_i 和 β_i 使得 $\theta_{\text{Primal}}(\boldsymbol{\omega})$ 为正无穷，当 $g_i(\boldsymbol{\omega})$ 和 $h_i(\boldsymbol{\omega})$ 满足条件时，则 $\theta_{\text{Primal}}(\boldsymbol{\omega})=f(\boldsymbol{\omega})$。这样，目标函数可以写为（原问题）：

$$\min_{\omega} \theta_{\text{Primal}}(\boldsymbol{\omega}) = \min_{\omega} \max_{\substack{\alpha:\alpha_i \geqslant 0 \\ \beta}} L(\boldsymbol{\omega}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{1.28}$$

这个过程通常不容易进行，所以考虑如下的问题（对偶问题）：

$$\max_{\substack{\alpha:\alpha_i \geqslant 0 \\ \beta}} \theta_{\text{Dual}}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \max_{\substack{\alpha:\alpha_i \geqslant 0 \\ \beta}} \min_{\omega} L(\boldsymbol{\omega}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{1.29}$$

仅仅更换了最大与最小的顺序，那么在什么条件下两者会等价？假设 f，g 都是凸函数，h 为仿射函数，求得的 $\boldsymbol{w}^*$，$\boldsymbol{\alpha}^*$，$\boldsymbol{\beta}^*$ 满足 KKT 条件，即

$$\begin{cases} \dfrac{\partial L(\boldsymbol{w}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{w}_i} = 0 \\ \dfrac{\partial L(\boldsymbol{w}^*, \boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)}{\partial \boldsymbol{\beta}_i} = 0 \\ \alpha_i^* g_i(\boldsymbol{w}^*) = 0 \\ g_i(\boldsymbol{w}^*) \leqslant 0, \quad \boldsymbol{\alpha}^* \geqslant 0 \end{cases} \tag{1.30}$$

则原问题与对偶问题等价（具体的例子可以参考 SVM 线性分类器）。

梯度下降法是一个一阶最优化算法，对于数据集 $\{\boldsymbol{x}_i, y_i\}_{i=1}^m$，输入与输出之间的模型为 $y=h_\theta(\boldsymbol{x})$，通过如下的优化目标函数求解参数：

$$\min_{\theta} J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x_i)-y_i)^2 + \lambda \cdot R(\theta) \tag{1.31}$$

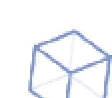
如何求解？通过初始化参数 $\theta^{(0)}$，然后通过上述目标函数关于参数的偏导数，求出梯度下降量（沿方向）：

$$\begin{cases} ?\theta = \left.\dfrac{\partial J(\theta)}{\partial \theta}\right|_{\theta=\theta^{(k)}} \\ \theta^{(k+1)} = \theta^{(k)} - \alpha \cdot ?\theta \end{cases} \tag{1.32}$$

更新直至损失函数满足一定的约束条件：将求得的 $\theta^{(k)}$ 代入至损失函数中，对于 $J(\theta^{(k)})$，若满足 $|J(\theta^{(k+1)})-J(\theta^{(k)})|\leqslant\varepsilon$，则退出。需要注意的是，如果目标函数为非凸优化的，则梯度下降法求得的并非全局最优解，可能会陷入局部最优。为了尽可能避免局部最优（鞍点、局部极值点），通常需要引入动量参数，修正下降的量，即 $\theta^{(k+1)}=\kappa\cdot\theta^{(k)}-\alpha\cdot ?\theta$，其中的 κ 为动量参数。

1.1.4 框架分析

小波分析、框架分析、多尺度几何分析是机器学习中应用最为广泛、理论基础最为清晰



的架构。首先,小波函数是指能够通过伸缩与平移生成平方可积空间 $L^2(\mathbb{R})$的一组基,该空间中的任意信号在该组基下的表示系数刻画该信号的时频局部化特性,即公式:

$$\begin{cases} \mathrm{WT}_f(a,b) = \int f(t) \cdot \overline{\psi_{a,b}(t)}\mathrm{d}t \\ \psi_{a,b}(t) = \dfrac{1}{\sqrt{|a|}}\psi\left(\dfrac{t-b}{a}\right) \end{cases} \tag{1.33}$$

其中,$\mathrm{WT}_f(a,b)$为小波系数,通过平移因子 b 和伸缩因子 a 所得到的函数簇构成 $L^2(\mathbb{R})$空间的一组基,即

$$\{\psi_{a,b}(t)\}_{a,b} \subseteq L^2(\mathbb{R}) \tag{1.34}$$

常用的一代小波函数包括 Doubechies 小波系列、Mayer 小波、Gaussian 小波以及 Morlet 小波等。需要注意的是,大多数良态小波的构造是基于多分辨分析(MRA),而且构造的小波是连续的;为了应用,通常将连续小波进行离散化操作,而且已经形成快速小波算法——Mallat 算法,包括分解公式与重构公式,其中分解公式为:

$$\begin{cases} a_{k+1}(n) = \sum\limits_m a_k(m) \cdot h(m-2n) \\ d_{k+1}(n) = \sum\limits_m a_k(m) \cdot g(m-2n) \end{cases} \tag{1.35}$$

重构公式为:

$$a_k(n) = \sum_m a_{k+1}(m) \cdot h(n-2m) + \sum_m d_{k+1}(m) \cdot g(n-2m) \tag{1.36}$$

式中 h,g 分别为低通滤波器(对应尺度函数)和高通滤波器(对应小波函数),符号“a,d”表示逼近成分和细节成分。

与小波分析相比,框架分析在信号处理中则更为常用,由框架带来的冗余性使得信号的表示具有多样性,可以在不同的约束下求解最优的表示系数,进而本质地刻画信号的特性。首先,框架不是基,是带有冗余特性(且线性相关)的向量组,例如简单的三维空间的基为:

$$\{(1\quad 0\quad 0),(0\quad 1\quad 0),(0\quad 0\quad 1)\} \subseteq \mathbb{R}^3$$

但要使基变为框架,便可以任意加该空间的向量,如加向量$(1\quad 1\quad 0)$,则得到的框架为:

$$\{(1\quad 0\quad 0),(0\quad 1\quad 0),(0\quad 0\quad 1),(1\quad 1\quad 0)\} \subseteq \mathbb{R}^3$$

所以该空间中的任意一个向量在该框架下的表示是不唯一的,而这个多种表示方式恰好是由框架的冗余性带来的。进一步,对于 $L^2(\mathbb{R})$空间中的一个框架,记为$\{\phi_{m,n}\}\subseteq L^2(\mathbb{R})$,注意,其线性组合具有自表示性,对于 $\forall f \in L^2(\mathbb{R})$,有:

$$A \cdot \|f\|_2^2 \leqslant \sum_{m,n} |\langle \phi_{m,n}, f\rangle|_2^2 \leqslant B \cdot \|f\|_2^2 \tag{1.37}$$

其中的 A,B 为框架下界与上界,通常由小波构造的框架称为小波框架。另外,需要注意的是框架理论是表示学习的数学基础。

多尺度几何分析是第二代小波系统,克服了第一代小波的缺点——点检测,代表性的第二代小波包括:脊波、曲波、楔形波、带状波、条状波、轮廓波等。相比之前的第一代小波,第二代小波大多数没有解析形式。在信号与信息处理过程中通常作为人工提取特征的工具,



为模式识别任务提供进一步的分析。近些年随着神经网络的广泛使用，开始将第一代小波作为激活函数引入到神经网络中形成小波神经网络，充分发挥其局部化、方向性、多尺度特性等优势，但第二代小波通常很难作为激活函数，所以其主要用于滤波器组的构造，然后初始化神经网络模型的参数，尽可能地避免过早陷入局部最优。

1.2 稀疏表示

本节给出稀疏学习的发展脉络以及有关稀疏的热点研究课题，并结合灵长类动物视觉皮层方面研究的进展，给出稀疏神经认知的发展历程。

1.2.1 稀疏表示初步

在信号与图像处理过程中，模型是至关重要的。借助于合适的模型，可以处理各种任务，如去噪、恢复、分离、内插、外插、压缩、采样、分析和合成、检测、识别等。

该模型的核心在于线性代数中研究的一个简单的欠定线性方程组。给一个满秩的矩阵 $\boldsymbol{A}\in\mathbb{R}^{n\times m}(n<m)$产生一个欠定的线性方程组 $\boldsymbol{A}\cdot\boldsymbol{x}=\boldsymbol{b}$，我们知道在 $\boldsymbol{b}$ 已知的时候，该方程的解具有无穷多个，然而我们感兴趣的是求最稀疏的一个解，即该解的非零项的个数最少。那么这个解是不是唯一的？如果是，在什么时候？如何在耗时最少情况下找到这个稀疏解？显然，对于稀疏模型而言，这些问题是我们处理实际问题的动力与理论基础。另外，该领域的研究工作也是对线性代数、优化、科学计算等知识的一种延伸。

1. 稀疏学习学什么

稀疏学习的任务主要有稀疏编码、字典学习。在回答这个问题之前，首先给出稀疏信号及字典等相关概念。

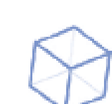
关于稀疏信号的定义，这里给出 4 种形式：严格 k 稀疏信号，可压缩信号，稀疏基下的稀疏信号，稀疏基下的可压缩信号。

(1) 严格 k 稀疏信号：考虑一个有限长信号 $\boldsymbol{x}\in\mathbb{R}^n$，如果信号 $\boldsymbol{x}$ 至多有 k 个非零元素，即 $\|\boldsymbol{x}\|_0\leqslant k$，则称信号 $\boldsymbol{x}$ 为严格 k 稀疏信号。

(2) 可压缩信号：如果信号可以用一个 k 稀疏向量来近似表示，则称这样的信号为可压缩性信号。

(3) 稀疏基下的稀疏信号：大多数的情况下，信号本身不是稀疏的，但是在某些合适的基或变换下稀疏，例如一个正弦信号不是稀疏的，但它的傅里叶变换是稀疏的，只包含一个非零值。或者定义为：如果一个信号至多有 k 个非零变换系数，则称该信号是 k 稀疏的。

(4) 稀疏基下的可压缩信号：给定值 k，信号 x 的最佳近似 k 项元素的线性组合为 $\hat{\boldsymbol{x}}_k=\sum_{i=0}^{k-1}\alpha(i)\cdot\boldsymbol{\psi}(i)$，称 $\hat{\boldsymbol{x}}_k$ 为 $\boldsymbol{x}$ 的最佳 k 稀疏近似。信号的压缩程度取决于系数 α 中所保留下的



元素个数。

关于字典的概念,一般来说,字典 $\boldsymbol{A}$ 来自信号空间的元素集,其线性组合可以表示或近似表示信号。在我们经常关注的稀疏学习任务中,往往要求字典是一个“扁矩阵”,也称为过完备字典。在实际应用中,这样的字典优于正交基已经得到验证。

2. 稀疏编码

有关稀疏编码的最早的文献可以追溯到 1996 年 BA. Olshausen 和 D. J field 的工作,他们考虑哺乳动物初级视觉皮层简单细胞的感受野的三个性质,即空间局部化、方向性和带通特性。如何理解感受野的这些性质,并在自然图像处理中得到应用呢? 一种已有的理解视觉神经元的反应性质的方法就是,考虑用有效编码的方式,将这些性质对应为自然图像的统计结构。沿着这个思路,大量的研究试图在自然图像上去训练无监督的训练方法,以获得类似于感受野的相似性质为目的。但是,没有一个能成功获得可以张成图像空间并包含上面三个性质的计算模型。BA. Olshausen 和 D. J field 的工作首次利用了极大化稀疏的特性去解释这些性质,他们的核心观点是:

$$E = -[\text{preserve inf}] - \lambda[\text{sparseness of } a_i] \tag{1.38}$$

其中的信息保持项可以写为:

$$[\text{preserve inf}] = -\sum_{x,y}\left[I(x,y) - \sum a_i \cdot \boldsymbol{\varphi}_i(x,y)\right]^2 \tag{1.39}$$

系数的稀疏特性则定义为:

$$[\text{sparseness of } a_i] = -\sum_i S\left(\frac{a_i}{\sigma}\right) \tag{1.40}$$

这里的 σ 是一个尺度常数,函数 $S(x)$的选择可以是$-\mathrm{e}^{-x^2}$,$\log(1+x^2)$或$|x|$等,这些选择都可以使得系数具有很少的非零系数。

其后沿着这个思路,我们考虑的稀疏编码问题可以归为求解如下的问题:

给定一个过完备的字典 $\boldsymbol{D}\in\mathbb{R}^{n\times m}(n<m)$以及一个信号 $x\in\mathbb{R}^n$,有:

$$P_0: \min_{\alpha} \|\boldsymbol{\alpha}\|_0 \quad \text{s.t.} \ \|\boldsymbol{x} - \boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2 \leqslant \varepsilon \tag{1.41}$$

这个问题是 NP 难的(组合优化问题),所以对上面这个问题求解的思路就有两种,分别为 1993 年 Stephane Mallat 和 Zhifeng Zhang 提出的贪婪算法和 1995 年 Scott Shaobin Chen、David Donoho 和 Michael Saunders 提出的松弛算法。下面分别就这两种求解的思路给出详细的介绍。

贪婪算法——首先我们介绍贪婪算法的核心观念:假设字典 $\boldsymbol{D}$ 满足 $\text{spark}(\boldsymbol{D})>2$,那么在求解的最优解中,非零项系数有 $\text{val}(P_0)=1$,进而我们需要找到这个解,可以利用:

$$\min\varepsilon(j) = \|\boldsymbol{a}_j\cdot z_j - \boldsymbol{b}\|_2 \xrightarrow{\text{得到}} z_j^* = \boldsymbol{a}_j^{\mathrm{T}}\cdot\boldsymbol{b}/\|\boldsymbol{a}_j\|_2^2 \tag{1.42}$$

进而得到:

$$\varepsilon(j) = \|\boldsymbol{b}\|_2^2 - \frac{(\boldsymbol{a}_j^{\mathrm{T}}\boldsymbol{b})^2}{\|\boldsymbol{a}_j\|_2^2} \tag{1.43}$$

如果哪一项 $\varepsilon(j)$ 最小，相应的 z_j^* 便是所要求得的非零系数项。利用相同的推理，假设 $\text{spark}(\boldsymbol{D})>2k_0$，那么我们知道 $\text{val}(P_0)=k_0$，这样为了找到这个解的非零项，需要枚举 $\binom{m}{k_0}$ 次从字典 $\boldsymbol{D}$ 得到 k_0 列，然后测试每一列，如果哪一项得到最小的稀疏解就是非零系数项。然而这个过程的时间复杂度为 $O(m^{k_0}nk_0)$，是非常耗时的。

因此，贪婪算法放弃了穷举式搜索，而支持局部最优单项更新，即初始的解为 $\boldsymbol{\alpha}^0=\boldsymbol{0}$，和相应的支撑集为空集，然后通过迭代更新，每一次增加一个非零项系数，直至第 k 次得到 $\boldsymbol{\alpha}^k$，其终止迭代的条件为 $\boldsymbol{r}^k=x-\boldsymbol{D}\cdot\boldsymbol{\alpha}^k$ 的二范数小于给定的 ε 为止。

代表的贪婪算法有：正交匹配追踪(Orthogonal Matching Pursuit，OMP)，匹配追踪(Matching Pursuit，MP)，弱匹配追踪(Weakly Matching Pursuit，WMP)和阈值算法(Threshold Algorithm，TA)。其中最为典型和常用的是 OMP 算法。

松弛算法——首先，松弛算法求解的问题是放松 P_0 问题中的 L_0 范数，通过利用连续或者光滑逼近它，通常松弛的方式包括 $L_p(p\in(0,1])$ 范数，或者为一些光滑函数 $\sum\log(1+\alpha x_j^2)$，$\sum x_j^2/(\alpha+x_j^2)$，或者 $\sum(1-\mathrm{e}^{-\alpha x_j^2})$ 等。

对于将 P_0 问题放松为如下的问题：

$$P_P:\min_{\alpha}\|\boldsymbol{\alpha}\|_p^p \quad \text{s.t.} \quad \boldsymbol{x}=\boldsymbol{D}\cdot\boldsymbol{\alpha} \tag{1.44}$$

如何求解这个问题？Gorodnitsky 和 Rao 提出了 FOCUSS 算法来求解此类问题，下面我们将其思想简单概述如下：

这种方法使用了迭代加权最小二乘(IRLS)将 $L_p(p\in(0,1])$ 范数表示为带有权值矩阵的 L_2 范数形式。在迭代求解的过程中，给定当前的解 $\boldsymbol{\alpha}_{k-1}$，权值矩阵设定为 $\boldsymbol{A}_{k-1}=\text{diag}(|\boldsymbol{\alpha}_{k-1}|^q)$，假设该矩阵是可逆的，则有

$$\|\boldsymbol{A}_{k-1}^{-1}\cdot\boldsymbol{\alpha}\|_2^2=\|\boldsymbol{\alpha}\|_{2-2q}^{2-2q} \tag{1.45}$$

进一步，设 $2-2q=p$，即变为 $\|\boldsymbol{\alpha}\|_p^p$。在实际中，我们通常不能保证权值矩阵是可逆的，所以通常取为伪逆，即 $\|\boldsymbol{A}_{k-1}^{\dagger}\cdot\boldsymbol{\alpha}\|_2^2$，基于此，我们利用拉格朗日乘子法来求解问题 P_p：

$$L(\boldsymbol{\alpha})=\|\boldsymbol{A}_{k-1}^{\dagger}\cdot\boldsymbol{\alpha}\|_2^2+\lambda^{\mathrm{T}}(\boldsymbol{x}-\boldsymbol{D}\cdot\boldsymbol{\alpha}) \tag{1.46}$$

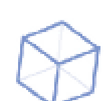
然后求导可以求得 $\boldsymbol{\alpha}_k$，迭代的停止准则是 $\|\boldsymbol{\alpha}_k-\boldsymbol{\alpha}_{k-1}\|_2$ 小于预先指定的阈值。

注：FOCUSS 算法是一种实际的策略，所得到的解是对于 P_0 问题的全局最优解的一种逼近。

另外一种松弛的策略是将 P_0 问题中的 L_0 范数直接变为 L_1 范数，必须注意字典中的原子是否进行了归一化是有一些小小的差别的。之前在 P_0 问题 L_0 范数与系数中的非零项是没有关系的，但是 $L_p(p\in(0,1])$ 范数趋于惩罚较大的非零项系数，为了避免这样的情况，我们应该对其进行合适的加权，新的问题就变为：

$$P_1:\min_{\alpha}\|\boldsymbol{W}^{-1}\cdot\boldsymbol{\alpha}\|_1 \quad \text{s.t.} \quad \boldsymbol{x}=\boldsymbol{D}\cdot\boldsymbol{\alpha} \tag{1.47}$$

其中，权值矩阵 $\boldsymbol{W}$ 的一种自然取法就是 $W(i,i)=1/\|\boldsymbol{d}_i\|_2$，如果字典是经过归一化处理过的，那么得到的矩阵 $\boldsymbol{W}=\boldsymbol{I}$，相应的解法就是 1995 年 Chen、Donoho 和 Saunders 提出的基匹



配算法(Basis Pursuit,BP)。

其次,在实际应用中,我们分析的问题是基匹配降噪(BPDN),由于已经假设字典 $\boldsymbol{D}$ 的原子经过归一化处理,求解的问题如下:

$$P_1^{\varepsilon}:\min_{\alpha}\|\boldsymbol{\alpha}\|_1 \quad \text{s.t.} \quad \|\boldsymbol{x}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2^2\leqslant\varepsilon \tag{1.48}$$

这个问题的求解,一方面可以利用线性规划去求解;另外一方面也可以通过迭代加权的最小二乘法来求解(Iterative-Reweighted-Least-Squares,IRLS)。前者已经可以通过各种优化软件进行求解,但是数据量较大的时候,二次规划的求解过程过于慢并且还需要对一些具体软件中的技术进行改进。我们关注 IRLS,它可以通过拉格朗日乘子将 P_1^{ε} 问题转化为下面的无约束优化问题:

$$Q_1^{\lambda}:\min_{\alpha}\lambda\|\boldsymbol{\alpha}\|_1+\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2^2 \tag{1.49}$$

注意,这里的 λ 是关于 $\boldsymbol{x},\boldsymbol{D},\varepsilon$ 的函数。通过设置 $\boldsymbol{\Lambda}=\mathrm{diag}(|\boldsymbol{\alpha}|)$,有 $\|\boldsymbol{\alpha}\|_1\equiv\boldsymbol{\alpha}^{\mathrm{T}}\cdot\boldsymbol{\Lambda}^{-1}\cdot\boldsymbol{\alpha}$,给定当前的一个逼近解 $\boldsymbol{\alpha}_{k-1}$,可以得到 $\boldsymbol{\Lambda}_{k-1}^{-1}$,我们可以求解:

$$M_k:\min_{\alpha}\lambda\boldsymbol{\alpha}^{\mathrm{T}}\cdot\boldsymbol{\Lambda}_{k-1}^{-1}\cdot\boldsymbol{\alpha}+\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2^2 \tag{1.50}$$

得到 $\boldsymbol{\alpha}_k$,不断更新直至满足停止准则 $\|\boldsymbol{\alpha}_k-\boldsymbol{\alpha}_{k-1}\|_2$。当然由于 λ 的不同,得到的解也不一样,如何选择 λ? 通常使用的方法就是最小角度回归(Least Angel Regression Stagewise,LARS),这种方法给出了随 λ 变化时,解 $\boldsymbol{\alpha}$ 中的每一项从零到非零变化的路径。λ 越小,解 $\boldsymbol{\alpha}$ 中的非零项的个数越多,反之越少。另外一种方法就是利用规范 L_2 差来绘制出随 λ 变化的曲线图,$\|\hat{\boldsymbol{x}}_{\lambda}-\boldsymbol{x}_0\|^2/\|\boldsymbol{x}_0\|^2$,选择其差最小时所对应的 λ。

最后,将此类问题转化为如下的形式:

$$f(\boldsymbol{\alpha})=\lambda\cdot\boldsymbol{1}^{\mathrm{T}}\cdot\rho(\boldsymbol{\alpha})+\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2^2 \tag{1.51}$$

其中,$\boldsymbol{1}$ 全为 1 的向量。

对于此问题形成了不同的迭代收缩算法,其中最为常用的 4 种算法为:可分替代函数法、基于迭代的最小二乘法、平行坐标下降法、逐阶段 OMP。

3. 字典学习

字典学习是稀疏模型中的核心,在信号与图像处理的过程中,如何针对应用场景和实际任务选择字典?一般而言,有预先指定字典、带参可调字典和学习字典三种方式。

预先指定字典——对于预先指定的字典,有离散余弦、非下采样小波、轮廓波、曲波等,这一类字典都有它所处理的具体的图像类,如图像中的 Cartoon 部分被认为是分段光滑且具有光滑的边界。一些预先指定的字典都有详尽的理论分析,估计表示系数的稀疏度,以此来简化信号的内容。

带参可调字典——在参数的控制下,可以通过调节参数来获得一组基或者框架,进而形成字典,其中最为熟知的就是小波包和带状波。例如小波包,可以通过计算信号在不同尺度



不同频带上的信息熵,进而选择最优的小波基来表示该信号。

注意:预先指定的字典或者带参可调字典具有快速的变换算法,所以计算的效率比较高,但是它们稀疏表示信号的能力有限。因此,大多数这类字典都被限制在特定的信号或者图像类,不适用于新的或者任意感兴趣的信号。

学习字典——为了避免前面两种字典稀疏表示能力限制的缺陷,通过学习的方式来获得字典。首先学习的前提是,需要建立信号样例的训练数据库,相似于在应用中所期望的信号;然后根据训练样例库构造一个经验学习字典,其中字典的原子是来自于经验数据,而不是一些理论模型;最后利用得到的字典对期望的信号进行处理。

学习字典具有以下两个特点:一是为了提升稀疏表示信号的能力,以较大的计算量为代价,使得学到的字典不具有清晰结构特性。另外一个学习的缺点是训练的方法被限制到低维信号上,这就是为什么处理图像的时候,需要在一些小的滑块上训练字典的原因。

学习字典的方法有:Engan 等人提出的最优方向法(Method of Direction,MOD)和 Aharon 等人 2006 年提出的 K-SVD(Kmeans-Singular Value Decomposition)方法。下面首先研究学习字典中的核心问题,然后给出经典的 K-SVD 方法学习字典的思路,之后总结 K-SVD 方法的缺点及改进的策略,最后介绍字典学习的最新进展。

我们所考察的问题是:

$$\min_{\boldsymbol{A},\{x_j\}_{i=1}^{M}} \sum_{i=1}^{M} \| y_i - \boldsymbol{A} \cdot x_i \|_2^2 \quad \text{s.t.} \ \| \boldsymbol{x}_i \|_0 \leqslant k_0, 1 \leqslant i \leqslant M \tag{1.52}$$

或者:

$$\min_{\boldsymbol{A},\{x_j\}_{i=1}^{M}} \sum_{i=1}^{M} \| \boldsymbol{x}_i \|_0 \quad \text{s.t.} \quad \| \boldsymbol{y}_i - \boldsymbol{A} \cdot \boldsymbol{x}_i \|_2^2 \leqslant \varepsilon, 1 \leqslant i \leqslant M \tag{1.53}$$

这个问题是否存在有意义的解?Aharon 等人回答了该问题,至少在 $\varepsilon=0$ 的时候,假设存在着一个字典 $\boldsymbol{A}_0$ 和一个充分多样的训练样例库,所有的样例可以由至多 k_0 个原子线性表示,则重新缩放和置换列原子,$\boldsymbol{A}_0$ 是唯一能够表示训练样例库中所有的样例的字典,即 K-SVD 算法。

为了得到字典和相应的稀疏表示系数,通过联合去求表示系数和字典。该算法包含两步,一是稀疏编码,即固定字典,利用 OMP 算法求解相应的稀疏表示系数;二是固定得到稀疏表示系数后来更新字典。这里主要陈述如何更新字典。

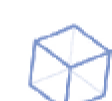
如上面的问题所示,共有 M 个训练样例,在固定字典的前提下,可以得到 M 个稀疏表示系数,将其按列排放得到一个矩阵,记为 $\boldsymbol{X}$;为了更新字典中的每一个原子,比如第 j_0 个原子,需要计算残差矩阵:

$$\boldsymbol{E}_{j_0} = \boldsymbol{Y} - \sum_{j \neq j_0} \boldsymbol{a}_j \cdot \boldsymbol{x}_j^{\mathrm{T}} \tag{1.54}$$

然后计算第 j_0 个原子所使用的支撑集:

$$\Omega_{j_0} = \{ i \mid \boldsymbol{X}(j_0, i) \neq 0, i = 1, 2, \cdots, M \} \tag{1.55}$$

之后计算残差矩阵在此支撑集上所对应的列,构成矩阵 $\boldsymbol{E}_{j_0}^R$,最后对此矩阵进行奇异值分



解，得到 $\boldsymbol{E}_{j_0}^R=\boldsymbol{U}\cdot\Delta\cdot\boldsymbol{V}^{\mathrm{T}}$，更新字典原子得到 $\boldsymbol{a}_{j_0}=\boldsymbol{u}_1$，以及表示系数：

$$\boldsymbol{x}_{j_0}^R=\Delta(1,1)\cdot\boldsymbol{v}_1 \tag{1.56}$$

K-SVD 算法的缺点和改进思路：

K-SVD 学习字典的思路比较简单，在实际中也获得了广泛的应用，但是它也有一些缺点，具体如下：

一是速度和记忆问题，与结构化的字典相比，训练得到的字典需要更多的计算量，因此使用和存储学习得到的字典，与传统的变换方法相比，往往缺乏有效性。

二是限制在低维信号，学习过程被限制在 $n\leqslant 1000$ 的低维信号上，超越这个维数会带来一系列的问题，如非常慢的学习的过程，过拟合的风险。

三是单尺度上的字典，不论是通过 MOD 还是 K-SVD 的算法训练得到的字典都是图像原本尺度上的考虑，但小波变换给我们的启示是信号在不同尺度上具有不同的信息量，能否构造多尺度上的字典？

四是缺乏不变量特性，在一些应用中，期望得到的字典具有一些不变量的性质，最为经典的性质就是平移不变性质、尺度不变性质，换言之，当字典应用在一幅平移/旋转/伸缩上的图像时，期望得到的稀疏表示与原始图像的表示具有相似性。

上面这些学习字典的缺点都是预先指定字典和带参调节字典的优点。下面针对上述缺点，提出一些学习字典的改进策略。

针对第二个缺点(限制在低维信号)，Ron Rubinstein 提出了稀疏 K-SVD 学习的策略，即双稀疏的算法。具体描述为：字典 $\boldsymbol{A}$ 中每一个原子能够表示为预先指定字典 $\boldsymbol{A}_0$ 的 k_0 个原子的线性组合，因此，能够写为 $\boldsymbol{A}=\boldsymbol{A}_0\cdot\boldsymbol{Z}$，这里的矩阵 $\boldsymbol{Z}$ 是一个每列只有 k_0 个非零项的稀疏矩阵。这样的选择有什么好处呢？这个字典 $\boldsymbol{A}$ 有快速运算的算法，因此利用 $\boldsymbol{A}$ 和它的伴随矩阵是比较容易的。之后，得到的求解问题如下：

$$\begin{aligned}&\min_{\{x_i\}_{i=1}^M,\{z_j\}_{j=1}^m}\sum_{i=1}^M\|\boldsymbol{y}_i-\boldsymbol{A}_0\cdot\boldsymbol{Z}\cdot\boldsymbol{x}_i\|_2^2\\&\text{s. t.}\begin{cases}\|\boldsymbol{z}_j\|_0\leqslant k_0,1\leqslant j\leqslant m\\\|\boldsymbol{x}_i\|_0\leqslant k_1,1\leqslant i\leqslant M\end{cases}\end{aligned} \tag{1.57}$$

如何求解式(1.57)？一方面，固定稀疏矩阵 $\boldsymbol{Z}$，利用 OMP 求解稀疏系数 $\{\boldsymbol{x}_i\}_{i=1}^M$；另外一方面固定稀疏系数 $\{\boldsymbol{x}_i\}_{i=1}^M$，更新稀疏矩阵 $\boldsymbol{Z}$。如何得到稀疏矩阵 $\boldsymbol{Z}$ 中每一个列或者每一个原子？只需要将上面的目标函数等价为：

$$\begin{aligned}\sum_i\|\boldsymbol{y}_i-\boldsymbol{A}_0\cdot\boldsymbol{Z}\cdot\boldsymbol{x}_i\|_2^2&=\left\|\boldsymbol{Y}-\boldsymbol{A}_0\cdot\sum_j\boldsymbol{z}_j\cdot\tilde{\boldsymbol{x}}_j^{\mathrm{T}}\right\|_F^2\\&=\|\boldsymbol{E}_j-\boldsymbol{A}_0\cdot\boldsymbol{z}_j\cdot\tilde{\boldsymbol{x}}_j^{\mathrm{T}}\|_F^2\end{aligned} \tag{1.58}$$

其中这里的 $\boldsymbol{E}_j=\boldsymbol{Y}-\boldsymbol{A}_0\sum_{k\neq j}\boldsymbol{z}_k\cdot\tilde{\boldsymbol{x}}_k^{\mathrm{T}}$，注意 $\tilde{\boldsymbol{x}}_j^{\mathrm{T}}$ 为稀疏系数矩阵 $\boldsymbol{X}$ 的第 j 行，之后根据：

$$\|\boldsymbol{E}_j-\boldsymbol{A}_0\cdot\boldsymbol{z}_j\cdot\tilde{\boldsymbol{x}}_j^{\mathrm{T}}\|_F^2=\|\boldsymbol{E}_j\cdot\tilde{\boldsymbol{x}}_j-\boldsymbol{A}_0\cdot\boldsymbol{z}_j\|_F^2+f(\boldsymbol{E}_j,\tilde{\boldsymbol{x}}_j) \tag{1.59}$$

得到等价的问题求解稀疏矩阵 $\boldsymbol{Z}$ 中的每一列，即



$$\min_{z_j} \| \boldsymbol{E}_j \cdot \tilde{\boldsymbol{x}}_j - \boldsymbol{A}_0 \cdot \boldsymbol{z}_j \|_2^2 \quad \text{s. t.} \quad \| \boldsymbol{z}_j \|_0 \leqslant k_0 \tag{1.60}$$

这样通过 OMP 算法便可以得到更新的稀疏矩阵 $\boldsymbol{Z}$。

针对第一个缺点，可以在学习的字典中嵌入一些结构，最简单的方法就是将多个酉矩阵合并起来形成一个过完备的字典，这种结构的字典能够保证得到的是紧框架，即它的伴随矩阵与伪逆是相等的。描述如下：为了简便，给出两个酉矩阵，将其合并为一个字典矩阵 $\boldsymbol{A}=[\boldsymbol{\Psi},\boldsymbol{\Phi}]\in\mathbb{R}^{n\times 2n}$，下面主要集中在字典的更新阶段，即目标是：

$$\min_{\boldsymbol{\Psi},\boldsymbol{\Phi}} \| \boldsymbol{\Psi} \cdot \boldsymbol{X}_\Psi + \boldsymbol{\Phi} \cdot \boldsymbol{X}_\Phi - \boldsymbol{Y} \|_F^2 \quad \text{s. t.} \quad \boldsymbol{\Psi}^{\mathrm{T}} \cdot \boldsymbol{\Psi} = \boldsymbol{\Phi}^{\mathrm{T}} \cdot \boldsymbol{\Phi} = \boldsymbol{I} \tag{1.61}$$

如何求解 $\boldsymbol{\Psi},\boldsymbol{\Phi}$？这里采用固定 $\boldsymbol{\Psi}$，更新 $\boldsymbol{\Phi}$，再固定 $\boldsymbol{\Phi}$，去更新 $\boldsymbol{\Psi}$；这样便可以利用著名的 Procrustes Problem，描述如下，求解：

$$\min_{\boldsymbol{Q}} \| \boldsymbol{A} - \boldsymbol{Q} \cdot \boldsymbol{B} \|_F^2 \quad \text{s. t.} \quad \boldsymbol{Q}^{\mathrm{T}} \cdot \boldsymbol{Q} = \boldsymbol{I} \tag{1.62}$$

可以将上面的目标函数写为：

$$\| \boldsymbol{A} - \boldsymbol{Q} \cdot \boldsymbol{B} \|_F^2 = \mathrm{tr}\{\boldsymbol{A}^{\mathrm{T}} \cdot \boldsymbol{A}\} + \mathrm{tr}\{\boldsymbol{B}^{\mathrm{T}} \cdot \boldsymbol{B}\} - 2\mathrm{tr}\{\boldsymbol{Q} \cdot \boldsymbol{B} \cdot \boldsymbol{A}^{\mathrm{T}}\} \tag{1.63}$$

那么极小化上式，转化为极大化 $\mathrm{tr}\{\boldsymbol{Q}\cdot\boldsymbol{B}\cdot\boldsymbol{A}^{\mathrm{T}}\}$，根据迹的性质，有：

$$\begin{aligned} &\mathrm{tr}\{\boldsymbol{Q} \cdot \boldsymbol{B} \cdot \boldsymbol{A}^{\mathrm{T}}\} \xrightarrow{\boldsymbol{B} \cdot \boldsymbol{A}^{\mathrm{T}} = \boldsymbol{U} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{V}^{\mathrm{T}}} \mathrm{tr}\{\boldsymbol{Q} \cdot \boldsymbol{U} \cdot \boldsymbol{\Sigma} \cdot \boldsymbol{V}^{\mathrm{T}}\} \xrightarrow{\mathrm{tr}(\boldsymbol{a} \cdot \boldsymbol{b}) = \mathrm{tr}(\boldsymbol{b} \cdot \boldsymbol{a})} \\ &\mathrm{tr}\{\boldsymbol{V}^{\mathrm{T}} \cdot \boldsymbol{Q} \cdot \boldsymbol{U} \cdot \boldsymbol{\Sigma}\} \xrightarrow{\boldsymbol{V}^{\mathrm{T}} \cdot \boldsymbol{Q} \cdot \boldsymbol{U} = \boldsymbol{Z}} \mathrm{tr}\{\boldsymbol{Z} \cdot \boldsymbol{\Sigma}\} = \sum_i \sigma_i z_{i,i} \leqslant \sum_i \sigma_i \end{aligned} \tag{1.64}$$

所以选择 $\boldsymbol{Q}=\boldsymbol{V}\cdot\boldsymbol{U}^{\mathrm{T}}$，这样 $\boldsymbol{Z}=\boldsymbol{I}$，便可以使得 $\mathrm{tr}\{\boldsymbol{Q}\cdot\boldsymbol{B}\cdot\boldsymbol{A}^{\mathrm{T}}\}$ 取极大值。

针对第四个缺点，为了使字典具有一定的不变量性质，Aharon 等提出了一种特征字典，这种字典的结构性质引入了平移不变性质，描述为：假设所需要的字典 $\boldsymbol{A}\in\mathbb{R}^{n\times m}$ 是由一单信号 $\boldsymbol{a}_0\in\mathbb{R}^{m\times 1}$ 构造形成的，通过提取所有长度为 n 的块（包括循环平移形成的块），称单信号为特征信号，它所定义的字典为特征字典。对于一个信号 $\boldsymbol{y}$，可以由这个字典来进行表示：

$$\boldsymbol{y} = \sum_{k=1}^{m} x_k \cdot \boldsymbol{a}_k = \sum_{k=1}^{m} x_k \cdot R_k \boldsymbol{a}_0 \tag{1.65}$$

这里的 R_k 为一个算子，从特征信号中的第 k 个位置提取长度为 n 的块。现在在此结构的帮助下，给出字典学习的目标：

$$\min_{\boldsymbol{A},\{\boldsymbol{x}_i\}_{i=1}^M} \sum_{i=1}^{M} \| \boldsymbol{y}_i - \boldsymbol{A} \cdot \boldsymbol{x}_i \|_2^2 \quad \text{s. t.} \quad \| \boldsymbol{x}_i \|_0 \leqslant k_0, \quad i = 1,2,\cdots,M \tag{1.66}$$

对于稀疏编码阶段，仍用 OMP 算法去求解；但是在字典更新阶段，可以进行如下求解：

$$\sum_{i=1}^{M} \| \boldsymbol{y}_i - \boldsymbol{A} \cdot \boldsymbol{x}_i \|_2^2 = \sum_{i=1}^{M} \left\| \boldsymbol{y}_i - \sum_{k=1}^{m} x_i(k) \cdot R_k \boldsymbol{a}_0 \right\|_2^2 \tag{1.67}$$

为了取极值，求其导数，可以得到：

$$\sum_{i=1}^{M} \left(\sum_{k=1}^{m} x_i(k) \cdot R_k \right)^{\mathrm{T}} \left(\boldsymbol{y}_i - \sum_{k=1}^{m} x_i(k) \cdot R_k \boldsymbol{a}_0 \right) = 0 \tag{1.68}$$

进而得到特征信号的最优表达式：



$$\boldsymbol{a}_0^{\text{opt}}=\left(\sum_{k=1}^{m}\sum_{j=1}^{m}\left[\sum_{i=1}^{M}x_i(k)\cdot x_i(j)\right]R_k^{\mathrm{T}}R_j\right)^{-1}\sum_{i=1}^{M}\sum_{k=1}^{m}x_i(k)\cdot R_k^{\mathrm{T}}\boldsymbol{y}_i \tag{1.69}$$

这个结构有哪些优点？一是由于字典的自由度远小于 mn，所以说仅利用较少训练数据便可以得到特征字典，另外由学习过程的快速收敛特性也可以得到，这种字典的适用性是具有平移特性的信号或者图像。二是这个结构字典中原子的尺寸容易调节。

针对第三个缺点，2007 年 Julien Mairal、Guillermo Spiro 和 Michael Elad 三人提出了多尺度字典学习的策略，这也是对 K-SVD 单尺度学习字典的一种改进。为了描述的方便，提出全局 K-SVD 字典关于降噪的问题：

$$\{\hat{\boldsymbol{\alpha}}_{i,j},\hat{\boldsymbol{D}},\hat{\boldsymbol{x}}\}=\arg\min\lambda\|\boldsymbol{x}-\boldsymbol{y}\|_2^2+\sum_{i,j}\mu_{i,j}\|\boldsymbol{\alpha}_{i,j}\|_0+\sum_{i,j}\|\boldsymbol{D}\cdot\boldsymbol{\alpha}_{i,j}-R_{i,j}\boldsymbol{x}\|_2^2 \tag{1.70}$$

关于这个问题的求解，包括稀疏编码、字典更新、重构信号。首先，如果字典 $\boldsymbol{D}$ 是已知的，那么未知量有两个，一个是稀疏表示系数$\hat{\boldsymbol{\alpha}}_{i,j}$；另一个是整体输出图像 x。接下来处理的思路是，令 $x=y$ 时，先利用稀疏编码求解如下的问题：

$$\hat{\boldsymbol{\alpha}}_{i,j}=\arg\min\mu_k\|\boldsymbol{\alpha}_{i,j}\|+\|\boldsymbol{D}\cdot\boldsymbol{\alpha}_{i,j}-R_{i,j}\boldsymbol{x}\|_2^2 \tag{1.71}$$

得到全部的稀疏表示系数$\{\hat{\boldsymbol{\alpha}}_{i,j}\}_{i,j}$，之后更新求解全局信号：

$$\hat{\boldsymbol{y}}=\arg\min_{\boldsymbol{y}}\lambda\|\boldsymbol{x}-\boldsymbol{y}\|_2^2+\sum_{i,j}\|\boldsymbol{D}\cdot\boldsymbol{\alpha}_{i,j}-R_{i,j}\boldsymbol{x}\|_2^2 \tag{1.72}$$

它具有一个闭形式的解，即

$$\hat{\boldsymbol{y}}=\left(\lambda\boldsymbol{I}+\sum_{i,j}R_{i,j}^{\mathrm{T}}R_{i,j}\right)^{-1}\left(\lambda\boldsymbol{y}+\sum_{i,j}R_{i,j}^{\mathrm{T}}\boldsymbol{D}\cdot\boldsymbol{\alpha}_{i,j}\right) \tag{1.73}$$

其次，如果字典是未知的，那么也可以通过 K-SVD 算法进行学习，即在稀疏编码中，给一个初始的字典，然后在字典与稀疏表示系数之间进行迭代，得到字典和系数后，再进行重构得到去噪后的图像。

为什么需要多尺度稀疏表示？由于一幅图像的信息是呈多尺度分布的，如果能够获取不同尺度上的字典来表征这些不同尺度上的信息，之后将这些多尺度上的信息进行融合处理，便能得到一种对原图像更好的逼近。Julien Mairal、Guillermo Spiro 和 Michael Elad 等人提出了利用图像四叉树的多尺度信息分布和 K-SVD 训练字典的方法，得到每一尺度上的字典。下面分为两步分来阐述这篇文章的思想，一是四叉树模型选择多尺度结构的信息；二是每一尺度上的稀疏编码、字典更新，以及最后多尺度上的信息重构信号。

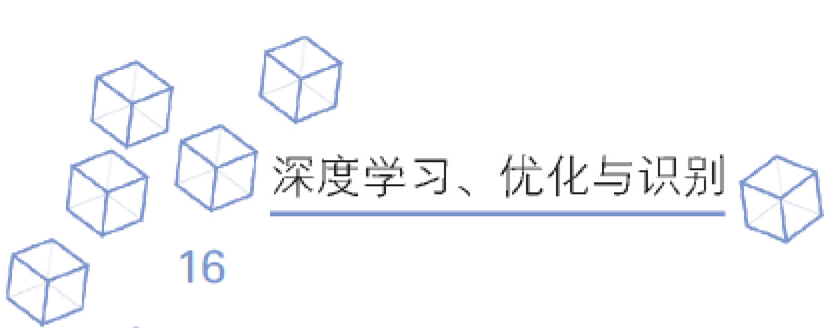
1）四叉树模型

给出多尺度上的四叉树模型如图 1.1 所示。

关于四叉树，有两个参数，一个是多尺度的个数 N；另外一个是树的深度。其关系通过 $n_s=n/4^s$ 来描述，其中，s 为尺度因子，n 为 $s=0$ 时滑块尺寸的大小，且有 $s=0,1,\cdots,N-1$。

2）稀疏编码、字典更新和重构信号

给出需要求解的问题表达式：



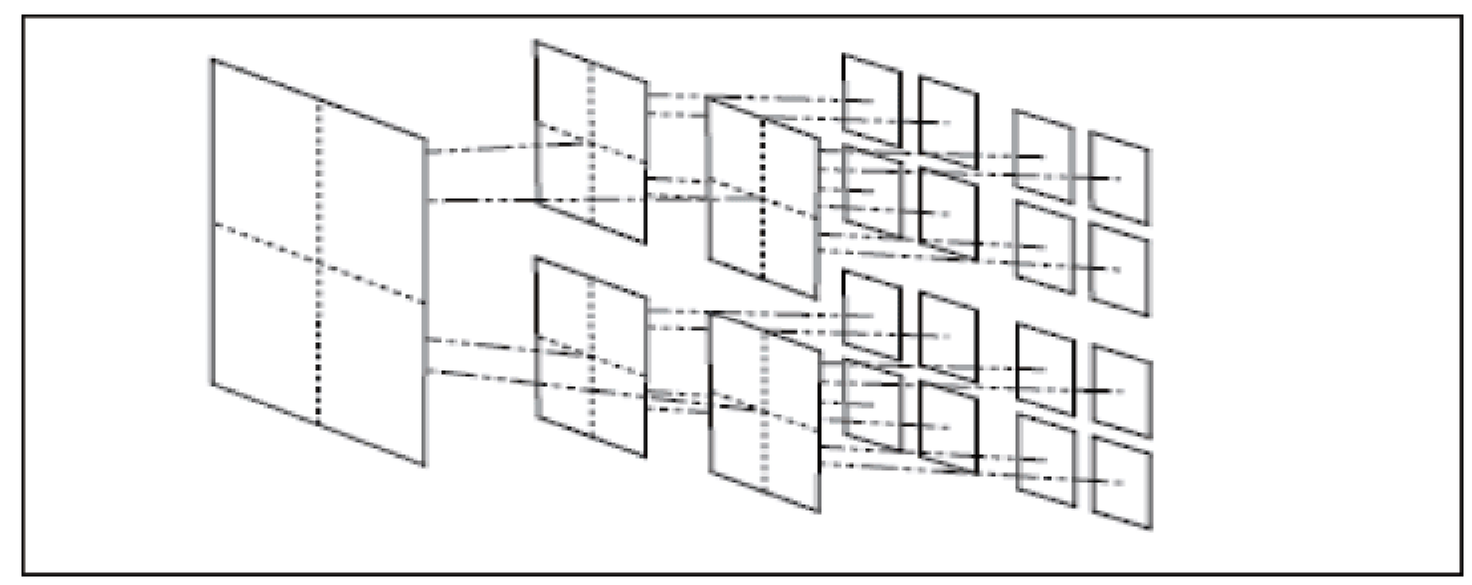


图 1.1　多尺度上的四叉树模型

$$\{\hat{\boldsymbol{y}},\hat{\boldsymbol{D}}_s,\hat{\boldsymbol{q}}_{s,k}^n\} = \arg\min \lambda \| \boldsymbol{x}-\boldsymbol{y} \|_2^2 + \sum_{s=0}^{N-1}\sum_{n=1}^{4^s}\sum_{k=1}^{M_s} \| \boldsymbol{D}_s \cdot \boldsymbol{q}_{s,k}^n - R_{s,k}^n \boldsymbol{x} \|_2^2 + \sum_{s=0}^{N-1}\sum_{n=1}^{4^s}\sum_{k=1}^{M_{s,n}} \mu_{s,k}^n \| \boldsymbol{q}_{s,k}^n \|_0 \tag{1.74}$$

符号解释：$\boldsymbol{D}_s$ 为尺度 s 上的字典；$\boldsymbol{q}_{s,k}^n$ 为尺度 s 上第 n 个位置上的第 k 个样本所对应的稀疏表示系数；$R_{s,k}^n$ 为从尺度 s 上第 n 个位置的图像上提取的第 k 个样本的算子。如何求解上述问题？仍然利用单尺度 K-SVD 的思路，分为每一尺度上的稀疏编码和字典更新，之后再进行重构。此处只给出已知的每一尺度字典和相应的表示系数上的重构公式，即考虑如下全局恢复问题：

$$\min \lambda \| \boldsymbol{x}-\boldsymbol{y} \|_2^2 + \sum_{s=0}^{N-1}\sum_{n=1}^{4^s}\sum_{k=1}^{M_s} \| \boldsymbol{D}_s \cdot \boldsymbol{q}_{s,k}^n - R_{s,k}^n \boldsymbol{x} \|_2^2 \tag{1.75}$$

最后，得到的恢复信号的闭形式解为：

$$\hat{\boldsymbol{y}} = \left(\lambda \boldsymbol{I} + \sum_s \sum_n \sum_k (R_{s,k}^n)^{\mathrm{T}} R_{s,k}^n\right)^{-1} \left(\lambda \boldsymbol{y} + \sum_s \sum_n \sum_k (R_{s,k}^n)^{\mathrm{T}} \boldsymbol{D}_s \cdot \boldsymbol{q}_{s,k}^n\right) \tag{1.76}$$

这个实验的结果在本书中不再赘述，具体可以参考 Julien Mairal、Guillermo Spiro 和 Michael Elad 的文章。

除了这种改进之外，2011 年 Boaz Ophir、Michael Lustig 和 Michael Elad 提出了利用小波变换的多尺度字典的学习策略。2007 年 Julien Mairal 等人的文章直接利用图像空域的四叉树模型的多尺度信息，Boaz Ophir 等人认为也可以利用其多尺度上的小波系数，通过对每一尺度上的小波系数进行学习来得到字典，之后处理相应尺度上的小波系数，再通过小波逆变换得到原始图像的一种逼近。简单地将这篇文章的思路描述如下：

首先建立一个训练样例图库，对其每一幅图像利用小波进行 N 尺度分解，前面每一尺度上，得到 3 个高频带，最后一个尺度上有 4 个频带，即一个低频带和 3 个高频带。通过收集所有图像，相同尺度和频带上的小波系数(尺度系数)，将其作为该尺度上的未处理的训练样例集，再进行滑块处理，得到该尺度上的训练样例集，利用 K-SVD 算法进行训练得到该尺度上的字典，这样便有 $3N+1$ 个字典。在测试阶段，给出一幅图像，假设其与训练样例库中的样例具有相似性(可以是噪声水平等)，通过同样的小波来进行相同尺度上的分解，对分

解后的每一个尺度上的小波系数利用相应尺度上的字典，利用 OMP 算法计算求解该小波系数的稀疏表示系数，之后得到小波系数的一个逼近，再通过逆小波变换得到原始图像的一个逼近。

4. 稀疏模型

接下来主要讨论信号处理中的稀疏模型，以及稀疏模型的最新进展。关于合成稀疏模型和分析稀疏模型的成果主要参考 Michael Elad 团队的工作。共分为三部分来论述，第一部分论述合成稀疏模型；第二部分分析稀疏模型；第三部分介绍稀疏模型的最新进展。在正式开始论述这些工作前，先给出一些已有的信号模型分析。

考虑样例集合 $\boldsymbol{Y}=\{\boldsymbol{y}_j\}_j\subseteq\mathbb{R}^n$，如果该样例集合是通过一幅图像滑块得到的，那么在一幅图像中，光滑的块是以较高的概率出现的，高度非光滑和失真的块是几乎不存在的。那么就可以利用贝叶斯框架下的概率密度函数给出描述先验分布 $P(\boldsymbol{y})$，先验在信号处理中已经得到了广泛的应用，如逆问题、压缩、异常检测等。例如考虑去噪问题，观测图像是由干净图像加噪声得到的，即 $\boldsymbol{y}=\boldsymbol{y}_0+\boldsymbol{n}$，已知噪声具有有限的能量 $\|\boldsymbol{n}\|_2\leqslant\varepsilon$，则优化问题变为：

$$\max_{\hat{\boldsymbol{y}}} P(\hat{\boldsymbol{y}}) \quad \text{s. t.} \quad \|\hat{\boldsymbol{y}}-\boldsymbol{y}\|_2\leqslant\varepsilon \tag{1.77}$$

对于先验表示形式的很多工作已经完成。下面给出两种常见的先验构造方式，其中一种最为常见的构造 $P(\boldsymbol{y})$ 的方式就是基于图像内容的直观期望。例如，吉布斯分布：

$$P(\boldsymbol{y}) = \text{Const}\cdot \mathrm{e}^{-\lambda\|\boldsymbol{L}\cdot\boldsymbol{y}\|_2^2} \tag{1.78}$$

其中 $\boldsymbol{L}$ 是拉普拉斯矩阵，$P(\boldsymbol{y})$ 是对图像 $\boldsymbol{y}$ 概率的一种评价。在这种先验中，光滑性被用于判断图像的概率，并且在信号与图像处理中得到广泛的应用。在这种先验的描述下，优化问题可以写为：

$$\min_{\hat{\boldsymbol{y}}} \|\boldsymbol{L}\cdot\hat{\boldsymbol{y}}\|_2^2 \quad \text{s. t.} \quad \|\hat{\boldsymbol{y}}-\boldsymbol{y}\|_2\leqslant\varepsilon \tag{1.79}$$

进一步利用拉格朗日乘子法，得到：

$$\min_{\hat{\boldsymbol{y}}} \|\boldsymbol{L}\cdot\hat{\boldsymbol{y}}\|_2^2+\mu\|\hat{\boldsymbol{y}}-\boldsymbol{y}\|_2^2 \tag{1.80}$$

因此它的解可以很容易得到：$\hat{\boldsymbol{y}}=\mu[\boldsymbol{L}^{\mathrm{T}}\cdot\boldsymbol{L}+\mu\boldsymbol{I}]^{-1}\boldsymbol{y}$，其中 μ 的选择应该满足问题中的限制 $\|\hat{\boldsymbol{y}}-\boldsymbol{y}\|_2\leqslant\varepsilon$。

与此类似的一个问题：如果信号 $\boldsymbol{y}=\boldsymbol{H}\cdot\boldsymbol{y}_0+\boldsymbol{n}$，其中 $\boldsymbol{H}$ 为线性退化算子，那么得到的解为 $\hat{\boldsymbol{y}}=\mu[\boldsymbol{L}^{\mathrm{T}}\cdot\boldsymbol{L}+\mu\boldsymbol{H}^{\mathrm{T}}\cdot\boldsymbol{H}]^{-1}\boldsymbol{y}$，这就是著名的 Wiener 滤波器。

由于直观上 L_1 范数要比 L_2 范数更为稀疏，所以近几年，将吉布斯分布中的 L_2 范数用 L_1 范数替代，得到：

$$\min_{\hat{\boldsymbol{y}}} \|\boldsymbol{L}\cdot\hat{\boldsymbol{y}}\|_1 \quad \text{s. t.} \quad \|\hat{\boldsymbol{y}}-\boldsymbol{y}\|_2\leqslant\varepsilon \tag{1.81}$$

这一选择类似于全变差算子(Total Variation，TV)。

另外一种构造先验的方法是基于信号的变换系数。例如，对于一个信号 $\boldsymbol{y}$，考察其小波变换 $\boldsymbol{T}$，得到小波系数 $\boldsymbol{T}\cdot\boldsymbol{y}$，在这种情况下，先验 PDF 就为：

$$P(\boldsymbol{y}) = \text{Const} \cdot e^{-\lambda \| \boldsymbol{T} \cdot \boldsymbol{y} \|_p^p} \tag{1.82}$$

此处的 $p \in (0,1]$是为了保证稀疏。下面的分析求解与上述吉布斯分布采取了相同的考虑方式。在后继的研究中，除小波变换外，还可以考虑离散余弦变换(Discrete Cosine Transform，DCT)、哈达玛变换(Walsh-Hadamard Transform，HT)、主成分分析(Principal Components Analysis，PCA)。例如基于主成分分析的先验可以写为一个多变量的高斯分布：

$$P(\boldsymbol{y}) = \text{Const} \cdot e^{-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{c})^{\mathrm{T}} \boldsymbol{R}^{-1} (\boldsymbol{y}-\boldsymbol{c})} \tag{1.83}$$

其中 $\boldsymbol{c} = \frac{1}{N}\sum_j \boldsymbol{y}_j$，$\boldsymbol{R}$ 为自相关矩阵，即 $\boldsymbol{R} = \frac{1}{N}\sum_j (\boldsymbol{y}_j - \boldsymbol{c})(\boldsymbol{y}_j - \boldsymbol{c})^{\mathrm{T}}$。

上面基于先验得到的模型，对于给定信号，通过先验概率 $P(\boldsymbol{y})$来评价比较容易，但是从服从该分布中获得随机采样是相对困难的。为了解决这个问题，人们开始研究稀疏模型(Sparse-Land)，稀疏模型有两种模式，即合成与分析，下面分别来考虑这两种模型。

合成稀疏模型——利用信号稀疏来定义稀疏基下的稀疏信号，在合成稀疏生成模型中，从稀疏表示系数$\boldsymbol{\alpha}$出发，随机选择一个基数为 k 的支撑集 T(如果字典的原子个数为 m，那么这种选择有$\binom{m}{k}$种)，然后利用字典 $\boldsymbol{D}$ 中对应支撑集的列形成 $\boldsymbol{D}_T$，与相应的$\boldsymbol{\alpha}_T$ 相乘，最终得到感兴趣的信号 $\boldsymbol{y}$。可以发现，此时该信号对应着子空间$\overline{\text{span}}\{\boldsymbol{d}_j : \boldsymbol{d}_j \in \boldsymbol{D}, j \in T\}$，如果字典中原子 $\boldsymbol{d}_i$ 的指标 $i \notin T$，那么可以将它从字典 $\boldsymbol{D}$ 中去掉，因为其不影响这个子空间。

通常的合成稀疏模型写为：

$$\hat{\boldsymbol{y}}_s = \boldsymbol{D} \cdot \arg\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_0 \quad \text{s.t.} \ \| \boldsymbol{y} - \boldsymbol{D} \cdot \boldsymbol{\alpha} \|_2 \leqslant \varepsilon \tag{1.84}$$

此模型求解不再赘述，在实际中的应用分为：

(1) 分析。给一个信号 $\boldsymbol{y}$，能否找到潜在的表示系数$\boldsymbol{\alpha}_0$？这个过程叫做原子分解，解决的问题为：

$$\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_0 \quad \text{s.t.} \ \| \boldsymbol{y} - \boldsymbol{D} \cdot \boldsymbol{\alpha} \|_2 \leqslant \varepsilon \tag{1.85}$$

通常将利用稀疏编码求得上述问题的解记为$\boldsymbol{\alpha}^*$，虽然它不一定是潜在的$\boldsymbol{\alpha}_0$，但是它是稀疏的，有较少的非零项。如果非零项的个数越少，那么$\boldsymbol{\alpha}^*$ 在以$\boldsymbol{\alpha}_0$ 为原点、半径为 ε 的圆的区域里出现的概率越大。一种现象：通常仿真实验中得到的$\boldsymbol{\alpha}^*$ 与$\boldsymbol{\alpha}_0$ 差得较大(支撑集都有误)，但是却不影响最后的信号重建效果。

(2) 逆问题。假设得到的直接观测为$\tilde{\boldsymbol{y}} = \boldsymbol{H} \cdot \boldsymbol{y} + \boldsymbol{n}$，这里的 $\boldsymbol{H}$ 为线性退化算子，$\boldsymbol{n}$ 为噪声或者扰动项，求解的问题为：

$$\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_0 \quad \text{s.t.} \ \| \tilde{\boldsymbol{y}} - \boldsymbol{H} \cdot \boldsymbol{D} \cdot \boldsymbol{\alpha} \|_2 \leqslant \varepsilon \tag{1.86}$$

可以得到原始信号 $\boldsymbol{y}$ 的逼近 $\boldsymbol{D} \cdot \boldsymbol{\alpha}^*$ 及其在字典下所对应的稀疏系数$\boldsymbol{\alpha}^*$。

(3) 压缩传感。对于一个稀疏信号，可以从较少的观测中更好地重建原信号，实际得到的直接观测为 $\boldsymbol{c} = \boldsymbol{P} \cdot \boldsymbol{y}$，这里的 $\boldsymbol{P} \in \mathbb{R}^{j_0 \times n}$ 为测量矩阵，$\boldsymbol{y}$ 为稀疏基下的 k 稀疏信号，其中 $j_0 \ll n$。通过求解问题：

$$\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_0 \quad \text{s.t.} \ \| \boldsymbol{c} - \boldsymbol{P} \cdot \boldsymbol{D} \cdot \boldsymbol{\alpha} \|_2 \leqslant \varepsilon \tag{1.87}$$

得到这个问题的解$\boldsymbol{\alpha}^*$，与字典$\boldsymbol{D}$相乘，其结果$\boldsymbol{D}\cdot\boldsymbol{\alpha}^*$尽可能为信号$\boldsymbol{y}$的条件是：传感矩阵$\boldsymbol{P}\cdot\boldsymbol{D}$满足限制等距条件(RIP)，换言之$j_0>2k$。

(4) 形态成分分析。假设观测得到的信号是两个子信号的叠加，即有$\boldsymbol{y}=\boldsymbol{y}_1+\boldsymbol{y}_2$，并且这两个子信号分别由两个稀疏模型产生，求解的问题为：

$$\min_{\boldsymbol{\alpha}}\|\boldsymbol{\alpha}_1\|_0+\|\boldsymbol{\alpha}_2\|_0 \quad \text{s.t.}\ \|\boldsymbol{y}-\boldsymbol{A}_1\cdot\boldsymbol{\alpha}_1-\boldsymbol{A}_2\cdot\boldsymbol{\alpha}_2\|_2\leqslant\varepsilon \tag{1.88}$$

通过求解上面的问题，便能得到看似合理的解$\hat{\boldsymbol{y}}_1=\boldsymbol{A}_1\cdot\boldsymbol{\alpha}^*$和$\hat{\boldsymbol{y}}_2=\boldsymbol{A}_2\cdot\boldsymbol{\alpha}^*$。

分析稀疏模型——相比于合成稀疏模型系统的研究，分析稀疏模型的研究相对比较“年轻”。这里关于分析稀疏模型的工作主要参考2011年S. Nam、M. E. Davies、M. Elad和R. Gribonval的文章。首先，给出一个信号是Cosparsity的定义：信号$\boldsymbol{x}\in\mathbb{R}^d$关于算子$\boldsymbol{\Omega}\in\mathbb{R}^{p\times d}$的Cosparsity定义为：

$$\text{Cosparsity}\quad l:=p-\|\boldsymbol{\Omega}\cdot\boldsymbol{x}\|_0 \tag{1.89}$$

另外，记$\Lambda=\{j:(\boldsymbol{\Omega}\cdot\boldsymbol{x})(j)=0,j=1,2,\cdots,p\}$为信号$\boldsymbol{x}$的Cosupport。

从这个分析的稀疏模型中可以看到该模型关注的是$\boldsymbol{\Omega}\cdot\boldsymbol{x}$的零项。分析稀疏模型如何生成信号$\boldsymbol{x}$？首先在算子$\boldsymbol{\Omega}$中随机地选取$l$行，并且记下其对应的指标集$\Lambda$，有$|\Lambda|=l$，然后随机形成一个信号$\boldsymbol{v}\in\mathbb{R}^d$，例如$\boldsymbol{v}$服从独立同分布的高斯概率密度函数。最后将信号$\boldsymbol{v}$正交投影到空间$(\overline{\text{span}}\{\boldsymbol{w}_j:\boldsymbol{w}_j\in\boldsymbol{\Omega},j\in\Lambda\})^{\perp}$，得到信号即$\boldsymbol{x}=(\boldsymbol{I}-\boldsymbol{\Omega}_\Lambda^{\mathrm{T}}(\boldsymbol{\Omega}_\Lambda\cdot\boldsymbol{\Omega}_\Lambda^{\mathrm{T}})^{-1}\boldsymbol{\Omega}_\Lambda)\boldsymbol{v}$。与合成稀疏模型中的分析类似，$\boldsymbol{x}$的Cosupport与算子定义了分析的子空间，即$(\boldsymbol{w}_j,\boldsymbol{x})=0,j\in\Lambda$或写为$\boldsymbol{x}\in(\overline{\text{span}}\{\boldsymbol{w}_j:\boldsymbol{w}_j\in\boldsymbol{\Omega},j\in\Lambda\})^{\perp}$，那么对于$(\boldsymbol{w}_j,\boldsymbol{x})\neq0$的算子$\boldsymbol{\Omega}$中，如果不考虑那些行，是不影响子空间的。

通常的分析稀疏模型写为：

$$\hat{\boldsymbol{x}}=\arg\min_{\boldsymbol{x}}\|\boldsymbol{\Omega}\cdot\boldsymbol{x}\|_0 \quad \text{s.t.}\quad \boldsymbol{y}=\boldsymbol{M}\cdot\boldsymbol{x} \tag{1.90}$$

其中$\boldsymbol{M}$为测量矩阵，注意这里$\|\boldsymbol{\Omega}\cdot\boldsymbol{x}\|_0\leqslant p-l$。如何求解这个问题？

贪婪分析算法(Greedy Analysis Pursuit：GAP)——如果信号x本身是严格$p-l$稀疏信号，可以取$\boldsymbol{\Omega}=\boldsymbol{I}$，这样便可以利用稀疏编码中的贪婪算法求解该问题，基于此，S. Nam、M. E. Davies、M. Elad和R. Gribonval提出了贪婪分析匹配算法(GAP)，这个算法的主要思想与合成贪婪算法中的阈值算法有一定的相似性，即首先给定输入$\boldsymbol{M}$,$\boldsymbol{\Omega}$,$\boldsymbol{y}$,l和$t\in(0,1]$，初次设置它的Cosupport集为$\Lambda_0=\{1,2,\cdots,p\}$，并求解：

$$\hat{\boldsymbol{x}}_0=\arg\min_{\boldsymbol{x}}\|\boldsymbol{\Omega}_{\Lambda_0}\cdot\boldsymbol{x}\|_2^2 \quad \text{s.t.}\quad \boldsymbol{y}=\boldsymbol{M}\cdot\boldsymbol{x} \tag{1.91}$$

得到的解如下：

$$\hat{\boldsymbol{x}}_0=\begin{bmatrix}\boldsymbol{\Omega}\cdot\boldsymbol{\Omega}^{\mathrm{T}}-\boldsymbol{M}^{\mathrm{T}}(\boldsymbol{M}\cdot\boldsymbol{M}^{\mathrm{T}})^{-1}\boldsymbol{M}\cdot\boldsymbol{\Omega}^{\mathrm{T}}\cdot\boldsymbol{\Omega}\\ \boldsymbol{M}\end{bmatrix}^{+}\begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{y}\end{bmatrix} \tag{1.92}$$

然后通过贪婪迭代，直至第k次，计算$\boldsymbol{\alpha}=\boldsymbol{\Omega}\cdot\hat{\boldsymbol{x}}_{k-1}$，得到$T_k=\{i:|\boldsymbol{\alpha}_i|\geqslant t\max_j|\boldsymbol{\alpha}_j|\}$，即避免$(\boldsymbol{w}_j,\boldsymbol{x})=0$的指标集合，然后去掉该支撑集合得到$\Lambda_k=\Lambda_{k-1}-T_k$，再利用：

$$\hat{\boldsymbol{x}}_k=\arg\min_{\boldsymbol{x}}\|\boldsymbol{\Omega}_{\Lambda_k}\cdot\boldsymbol{x}\|_2^2 \quad \text{s.t.}\quad \boldsymbol{y}=\boldsymbol{M}\cdot\boldsymbol{x} \tag{1.93}$$

更新得到解$\hat{\boldsymbol{x}}_k$，直至该迭代过程满足$k\geqslant p-l$时，终止迭代，输出结果。

当然从实际角度考虑，通常考察如下的分析稀疏模型，即

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \| \boldsymbol{\Omega} \cdot \boldsymbol{x} \|_0 \quad \text{s.t.} \quad \| \boldsymbol{y} - \boldsymbol{M} \cdot x \|_2 \leqslant \varepsilon \tag{1.94}$$

仍然利用上面 GAP 的思路，只是将其中的求解问题变为：

$$\hat{\boldsymbol{x}}_k = \arg\min_{x} \| \boldsymbol{\Omega}_{\Lambda_k} \cdot \boldsymbol{x} \|_2^2 \quad \text{s.t.} \quad \| \boldsymbol{y} - \boldsymbol{M} \cdot \boldsymbol{x} \|_2 \leqslant \varepsilon \tag{1.95}$$

这个通常利用拉格朗日乘子的方法进行求解，即可以写为：

$$\begin{aligned}\hat{\boldsymbol{x}}_k &= \arg\min_{x} \{ \| \boldsymbol{\Omega}_{\Lambda_k} \cdot \boldsymbol{x} \|_2^2 + \lambda \| \boldsymbol{y} - \boldsymbol{M} \cdot \boldsymbol{x} \|_2^2 \} \\ &= \arg\min_{x} \left\{ \left\| \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{bmatrix} - \begin{bmatrix} \boldsymbol{M} \\ \sqrt{\lambda}\, \boldsymbol{\Omega}_{\Lambda_k} \end{bmatrix} \cdot \boldsymbol{x} \right\|_2^2 \right\}\end{aligned} \tag{1.96}$$

其中 λ 为拉格朗日乘子，得到的解为：

$$\hat{\boldsymbol{x}}_k = \begin{bmatrix} \boldsymbol{M} \\ \sqrt{\lambda}\, \boldsymbol{\Omega}_{\Lambda_k} \end{bmatrix}^{+} \cdot \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{bmatrix} \tag{1.97}$$

凸松弛分析算法——基于凸松弛算法，得到需要求解的问题为：

$$\hat{\boldsymbol{x}} = \arg\min_{x} \| \boldsymbol{\Omega} \cdot \boldsymbol{x} \|_1 \quad \text{s.t.} \quad \| \boldsymbol{y} - \boldsymbol{M} \cdot \boldsymbol{x} \|_2 \leqslant \varepsilon \tag{1.98}$$

这个问题的理论工作可以参考 Candies 等人的经典结果，即如果测量矩阵 $\boldsymbol{M}$ 满足带有常数 $\delta_s^{\boldsymbol{\Omega}}$ 的 $\boldsymbol{\Omega}$-RIP 条件，其中 s 为信号 $\boldsymbol{x}$ 的稀疏度，那么由该问题求得的解与真实的解的关系满足：

$$\| \boldsymbol{x}^* - \boldsymbol{x} \| \leqslant C_0 \varepsilon + C_1 \frac{\| \boldsymbol{\Omega}^{\mathrm{T}} \cdot \boldsymbol{x} - (\boldsymbol{\Omega}^{\mathrm{T}} \cdot \boldsymbol{x})_s \|_1}{\sqrt{s}} \tag{1.99}$$

对于求解可以利用迭代收缩算法。

1.2.2　稀疏模型

关于稀疏模型的应用，分为合成稀疏模型和分析稀疏模型的应用。其中关于合成稀疏模型的应用主要有分析、降噪、逆问题、压缩传感、形态成分分析等；而关于分析稀疏模型的应用主要有压缩传感等。下面分别来介绍这些应用。

1. 合成模型的应用

描述下面应用的前提是合成稀疏模型的生成模型为 $M(\boldsymbol{A}, k_0, \boldsymbol{\alpha}, \varepsilon)$，并且该模型的参数已知。

(1) 分析：若 $\boldsymbol{y} \in M(\boldsymbol{A}, k_0, \boldsymbol{\alpha}, \varepsilon)$，那么能否找到 y 在字典 $\boldsymbol{A}$ 下的潜在的表示系数 $\boldsymbol{x}_0$？这个过程也被称为原子分解。显然，潜在的表示系数 $\boldsymbol{x}_0$ 服从 $\| \boldsymbol{A} \cdot \boldsymbol{x}_0 - \boldsymbol{y} \|_2 \leqslant \varepsilon$，但是除了 $\boldsymbol{x}_0$，还有很多的表示系数 $\boldsymbol{x}$ 也能使 $\| \boldsymbol{A} \cdot \boldsymbol{x} - \boldsymbol{y} \|_2 \leqslant \varepsilon$。求解的问题是：

$$P_0^{\varepsilon}: \quad \min_{x} \| \boldsymbol{x} \|_0 \quad \text{s.t.} \quad \| \boldsymbol{y} - \boldsymbol{A} \cdot \boldsymbol{x} \|_2 \leqslant \varepsilon \tag{1.100}$$

利用贪婪算法求解得到 $\hat{\boldsymbol{x}}_0$，虽然它与 $\boldsymbol{x}_0$ 不一样，甚至支撑集的差异性很大，但是这也不影响对于信号 $\boldsymbol{y}$ 的逼近性能。

（2）降噪：假设 $\boldsymbol{y}\in M(\boldsymbol{A},k_0,\boldsymbol{\alpha},\varepsilon)$ 为真实信号，但是由于在观测中引入了噪声 $\boldsymbol{n}$，并且知道噪声的能量，那么实际得到的观测信号为 $\tilde{\boldsymbol{y}}=\boldsymbol{y}+\boldsymbol{n}$。将信号 $\tilde{\boldsymbol{y}}$ 代入分析中的待求解问题，得到的解为 $\boldsymbol{x}_0^{\varepsilon+\delta}$。如果 k_0 很小，那么解 $\boldsymbol{x}_0^{\varepsilon+\delta}$ 在潜在的解 $\boldsymbol{x}_0$ 的 $\varepsilon+\delta$ 邻域内 $U(\boldsymbol{x}_0,\varepsilon+\delta)$，进而可以得到信号 $\boldsymbol{y}$ 的一个逼近 $\boldsymbol{A}\cdot\boldsymbol{x}_0^{\varepsilon+\delta}$。

（3）逆问题：假设观测得到的信号 $\tilde{\boldsymbol{y}}=\boldsymbol{H}\cdot\boldsymbol{y}+\boldsymbol{n}$，这里的线性算子能够表示模糊、投影、下采样，或者各种线性退化了的算子，$\boldsymbol{n}$ 为之前的噪声，求解的问题变为：

$$\min_{x}\|\boldsymbol{x}\|_0\quad \text{s.t.}\quad \|\tilde{\boldsymbol{y}}-\boldsymbol{H}\cdot\boldsymbol{A}\cdot\boldsymbol{x}\|_2\leqslant\varepsilon \tag{1.101}$$

得到的解为 $\boldsymbol{x}_0^{\varepsilon+\delta}$，然后与字典 $\boldsymbol{A}$ 相乘得到真实信号 $\boldsymbol{y}$ 的一个逼近。

（4）压缩传感：给定 $\boldsymbol{y}\in M(\boldsymbol{A},k_0,\boldsymbol{\alpha},\varepsilon)$，假设利用观测矩阵 $\boldsymbol{P}$ 得到的观测信号为 $\boldsymbol{c}=\boldsymbol{P}\cdot\boldsymbol{y}$，此处 $\boldsymbol{P}$ 可以是随机观测或者确定性观测的。无论哪一种观测，该观测矩阵与字典 $\boldsymbol{A}$ 相乘得到的传感矩阵 $\boldsymbol{D}^{CS}=\boldsymbol{P}\cdot\boldsymbol{A}$ 需要满足等距限制条件（Restricted Isometry Property，RIP）。然后求解问题：

$$\min_{x}\|\boldsymbol{x}\|_0\quad \text{s.t.}\quad \|\boldsymbol{c}-\boldsymbol{P}\cdot\boldsymbol{A}\cdot\boldsymbol{x}\|_2\leqslant\varepsilon \tag{1.102}$$

得到的解为 $\boldsymbol{x}_0^{\varepsilon}$，进而求出真实信号 $\boldsymbol{y}$ 的一个逼近 $\boldsymbol{A}\cdot\boldsymbol{x}_0^{\varepsilon}$。

（5）形态成分分析：给定信号 $\boldsymbol{y}_1\in M_1(\boldsymbol{A}_1,k_1,\boldsymbol{\alpha}_1,\varepsilon_1)$ 和 $\boldsymbol{y}_2\in M_2(\boldsymbol{A}_2,k_2,\boldsymbol{\alpha}_2,\varepsilon_2)$，观测信号为 $\boldsymbol{y}=\boldsymbol{y}_1+\boldsymbol{y}_2$，然后通过求解下面的问题：

$$P_0^{\varepsilon}:\quad \min_{x}\|\boldsymbol{x}_1\|_0+\|\boldsymbol{x}_2\|_0\quad \text{s.t.}\quad \|\boldsymbol{y}-\boldsymbol{A}_1\cdot\boldsymbol{x}_1-\boldsymbol{A}_2\cdot\boldsymbol{x}_2\|_2\leqslant\varepsilon_1+\varepsilon_2 \tag{1.103}$$

其解为 $(\boldsymbol{x}_1^{\varepsilon},\boldsymbol{x}_2^{\varepsilon})$，乘以相应的字典得到信号 $\boldsymbol{y}$ 的分离信号。作为恢复过程中的一部分，图像中逐段光滑的内容（Cartoon）和纹理部分必须分开考虑，此时形态成分分析就是必要的。

2. 分析模型的应用

描述下面分析模型的信号生成模型为 $M(\boldsymbol{\Omega},l,\boldsymbol{\alpha})$，即随机从分析算子 $\boldsymbol{\Omega}\in\mathbb{R}^{p\times d}$ 中抽出 l 行并记下相应的位置组成支撑集 $\boldsymbol{\Lambda}$，利用这 l 行线性张成一个空间 $W=\overline{\text{span}}\{\boldsymbol{w}_i:\boldsymbol{w}_i\in\boldsymbol{\Omega},i\in\Lambda\}$；然后随机形成一个信号 $\boldsymbol{v}\in\mathbb{R}^d$——服从独立同分布的高斯概率密度函数的信号，其中 $\boldsymbol{\alpha}$ 是相应的方差；最后将信号 $\boldsymbol{v}$ 正交投影到 W 的补空间上得到信号 $\boldsymbol{y}=(\boldsymbol{I}-\boldsymbol{\Omega}_\Lambda^{\mathrm{T}}(\boldsymbol{\Omega}_\Lambda\cdot\boldsymbol{\Omega}_\Lambda^{\mathrm{T}})^{-1}\boldsymbol{\Omega}_\Lambda)\boldsymbol{v}$，即 $\boldsymbol{y}\in M(\boldsymbol{\Omega},l,\boldsymbol{\alpha})$。

之前已经知道，从框架角度来讲，任意一个合成稀疏模型都有一个等价的分析模型，所以合成模型中的应用都对应着一个相应的分析模型的应用。下面来分析压缩传感。

给定信号 $\boldsymbol{y}\in M(\boldsymbol{\Omega},l,\boldsymbol{\alpha})$，通过观测矩阵 $\boldsymbol{P}$ 得到观测信号 $\boldsymbol{g}=\boldsymbol{P}\cdot\boldsymbol{y}$，实际中由于观测过程中引入的干扰项，导致实际观测的信号为 $\tilde{\boldsymbol{g}}=\boldsymbol{g}+\boldsymbol{n}$，其中干扰项的能量为 $\|\boldsymbol{n}\|_2\leqslant\varepsilon$。所以求解的问题为：

$$\min_{y}\|\boldsymbol{\Omega}\cdot\boldsymbol{y}\|_0\quad \text{s.t.}\quad \|\tilde{\boldsymbol{g}}-\boldsymbol{P}\cdot\boldsymbol{y}\|_2\leqslant\varepsilon \tag{1.104}$$

求解得到信号 $\boldsymbol{y}$ 的一个逼近 $\hat{\boldsymbol{y}}$。这样利用分析模式的压缩传感，可以将合成模型中的压缩传感的传感矩阵 $\boldsymbol{D}^{CS}=\boldsymbol{P}\cdot\boldsymbol{A}$ 分为观测矩阵 $\boldsymbol{P}$ 和字典 $\boldsymbol{A}$，因为之前的 $\boldsymbol{D}^{CS}$ 较满足 RIP 条件，而分开后，则只需要验证观测矩阵 $\boldsymbol{P}$ 是否满足 $\boldsymbol{A}$-RIP 条件即可，那么这里所使用的分析算子 $\boldsymbol{\Omega}$

为字典(框架)$\boldsymbol{A}$ 的对偶(框架)字典。

3. 基于稀疏模型在分类中的应用

首先，介绍 2009 年 Wright、Ma Yi 等人的工作，即稀疏表示分类(Sparse Representation for Classification，SRC)的思想，假设给定一个样例集合，分为训练和测试部分，训练样例集记为$\{(\boldsymbol{x}_i^{\text{Train}},y_i)\}_{i=1}^{M}$，其中 $\boldsymbol{x}_i^{\text{Train}}$ 为训练样本，y_i 为相应的类标；测试样例集记为$\{\boldsymbol{x}_n^{\text{Test}}\}_n$。对于训练样例集，将每一类的训练样本放在一起，如第 j 类就为：

$$X_j=\{\boldsymbol{x}_i^{\text{Train}}:y_i=j,i=1,2,\cdots,M\} \tag{1.105}$$

这里不妨取 $j=1,2,\cdots,N$，即有 N 类。然后对于每一类的训练样例集 X_j，将其中的样例按列排，形成一个矩阵记为 $\boldsymbol{D}_j$，也称为第 j 类的字典；这样便可以得到 N 个字典，将其级联形成一个大字典 $\boldsymbol{D}=[\boldsymbol{D}_1,\boldsymbol{D}_2,\cdots,\boldsymbol{D}_N]$。之后在测试阶段，需要通过求解下面的问题来得到每一个测试样本的类标：

$$\min_{x}\|\boldsymbol{\alpha}\|_0 \quad \text{s.t.} \quad \|\boldsymbol{x}_n^{\text{Test}}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2\leqslant\varepsilon \tag{1.106}$$

求解得到稀疏表示系数$\boldsymbol{\alpha}_n$，进一步可以得到逼近的信号$\hat{\boldsymbol{x}}_n^{\text{Test}}$，以及对应每一个子字典可以得到$\hat{\boldsymbol{x}}_j=\boldsymbol{D}_j\cdot\boldsymbol{\alpha}_{n,j}$，通过判断：

$$j_n=\arg\min_{1\leqslant j\leqslant N}\|\hat{\boldsymbol{x}}_n^{\text{Test}}-\hat{\boldsymbol{x}}_j\| \tag{1.107}$$

进而将测试样本 $\boldsymbol{x}_n^{\text{Test}}$ 分到第 j_n 类。

其次，介绍 2010 年 Qiang Zhang、Baoxin Li 的工作，他们的工作动机是将判别准则加入到稀疏模型中，使形成的新模型一方面具有稀疏表示能力，另外一方面也具有判别能力。此处的判别准则包括 Softmax 判别代价函数、费舍尔判别准则、线性预测分类误差、Logistic 代价函数等。下面简要地描述文章的思想：假设对于一个样例集，其中训练样例集记为$\{(\boldsymbol{x}_i^{\text{Train}},y_i)\}_{i=1}^{M}$，测试样例集记为$\{\boldsymbol{x}_n^{\text{Test}}\}_n$。将训练样本按列形成一个矩阵 $\boldsymbol{X}^{\text{Train}}$，相应的类标构成类标矩阵 $\boldsymbol{H}=[\boldsymbol{h}_1,\boldsymbol{h}_2,\cdots,\boldsymbol{h}_M]$，其中 $\boldsymbol{h}_i=(0,0,\cdots,1,\cdots,0)^{\text{T}}\in\mathbb{R}^N$ 是训练样例 $\boldsymbol{x}_i^{\text{Train}}$ 的类标，其第 i 个元素为 1，其他元素为 0，N 为类别的个数。通过求解下面的问题：

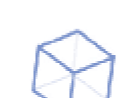
$$\begin{aligned}\{\hat{\boldsymbol{D}},\hat{\boldsymbol{W}},\hat{\boldsymbol{\alpha}}\}&=\arg\min_{\boldsymbol{D},\boldsymbol{W},\boldsymbol{\alpha}}\|\boldsymbol{X}^{\text{Train}}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2+\gamma\|\boldsymbol{H}-\boldsymbol{W}\cdot\boldsymbol{\alpha}\|\\ \text{s.t.}\quad&\|\boldsymbol{\alpha}_i\|_0\leqslant T,\quad i=1,2,\cdots,M\end{aligned} \tag{1.108}$$

其中 $\boldsymbol{D}$ 为稀疏表示字典，$\boldsymbol{W}$ 为判别能力字典，并且上面的第二项 $\|\boldsymbol{H}-\boldsymbol{W}\boldsymbol{\alpha}\|_2$ 为分类误差。通过交替迭代求解上面的优化问题，得到的解为$\{\hat{\boldsymbol{D}},\hat{\boldsymbol{W}}\}$。然后在测试阶段，对于测试信号 $\boldsymbol{x}_n^{\text{Test}}$，利用判别能力字典，得到 $\boldsymbol{l}=\boldsymbol{W}\cdot\boldsymbol{x}_n^{\text{Test}}\in\mathbb{R}^N$，通过判断：

$$j_n=\arg\max_{1\leqslant j\leqslant N}\boldsymbol{l} \tag{1.109}$$

进而将测试样本 $\boldsymbol{x}_n^{\text{Test}}$ 分到第 j_n 类。

基于分析稀疏模型的过参变量问题，首先从一个简单的例子开始，如果信号 $\boldsymbol{f}$ 是分段线性的，那么显然在每一段自变量取值内，对应的系数参数是常值。不妨设 $\boldsymbol{f}$ 分为$k+1$ 段，那么该函数具有 k 个变点位置，下面对该信号采集 d 个点，那么就有：



$$f = [\boldsymbol{I}, \boldsymbol{X}]\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{bmatrix} \tag{1.110}$$

其中 $\boldsymbol{I}$ 为单位阵，$\boldsymbol{X}=\mathrm{diag}(1,2,\cdots,d)$，$\boldsymbol{a},\boldsymbol{b}\in\mathbb{R}^d$ 为系数向量，此时 $\boldsymbol{a},\boldsymbol{b}$ 在有限差分算子 $\boldsymbol{\Omega}_{\mathrm{DIF}}$ 下是联合稀疏的，即 $\boldsymbol{\Omega}_{\mathrm{DIF}}\cdot\boldsymbol{a}$ 与 $\boldsymbol{\Omega}_{\mathrm{DIF}}\cdot\boldsymbol{b}$ 的非零位置相同。由于采样信号的长度为 d，而参数的个数却有 $2\cdot d$，所以这就是确定过参变量的问题。通过上述分析，得到如下的最小化问题：

$$\min_{\boldsymbol{a},\boldsymbol{b}}\left\|\boldsymbol{g}-[\boldsymbol{I},\boldsymbol{X}]\begin{pmatrix} \boldsymbol{a} \\ \boldsymbol{b} \end{pmatrix}\right\|_2^2 \quad \text{s. t.} \quad \|\, |\boldsymbol{\Omega}_{\mathrm{DIF}}\cdot\boldsymbol{a}|^2+|\boldsymbol{\Omega}_{\mathrm{DIF}}\cdot\boldsymbol{b}|^2 \,\| \leqslant k \tag{1.111}$$

注意这里的 $\boldsymbol{g}=\boldsymbol{f}+\boldsymbol{n}$，即 $\boldsymbol{g}$ 是由观测得到的，由真实信号干扰或者染噪造成。如果求解上面的优化问题得到的解为 $(\boldsymbol{a}^*,\boldsymbol{b}^*)$，那么便有原始信号的逼近：

$$\hat{f} = [\boldsymbol{I}, \boldsymbol{X}]\begin{bmatrix} \boldsymbol{a}^* \\ \boldsymbol{b}^* \end{bmatrix} \tag{1.112}$$

下面将该例进行推广，在推广过程中，描述得到的观测信号 $\boldsymbol{g}$ 为 $\boldsymbol{g}=\boldsymbol{M}\cdot\boldsymbol{f}+\boldsymbol{n}$，其中 $\boldsymbol{M}$ 为测量矩阵。将待求解的问题描述为：

$$\min_{\{\boldsymbol{a}_i\}}\left\|\sum_i |\boldsymbol{\Omega}_{\mathrm{DIF}}\cdot\boldsymbol{a}_i|^2\right\|_0 \quad \text{s. t.} \quad \left\|\boldsymbol{g}-\boldsymbol{M}\cdot[\boldsymbol{X}_1,\cdots,\boldsymbol{X}_n]\cdot\begin{pmatrix} \boldsymbol{a}_1 \\ \vdots \\ \boldsymbol{a}_n \end{pmatrix}\right\|_2^2 \leqslant \varepsilon \tag{1.113}$$

ε 是噪声的能量，即 $\|\boldsymbol{n}\|_2\leqslant\varepsilon$，$\boldsymbol{\Omega}$ 是一般的算子。如何求解这个问题？利用 GAP 算法的思想，对于给定输入 $[\boldsymbol{X}_1,\cdots,\boldsymbol{X}_n]$，$\boldsymbol{M}$，$\boldsymbol{\Omega}$，$\boldsymbol{g}$ 以及 $\sum_i |\boldsymbol{\Omega}\cdot\boldsymbol{a}_i|^2$ 的稀疏度 $p-l$，首先初始化 Cosupport $\Lambda_0=\{1,2,\cdots,p\}$，然后求解在 Λ_0 下的问题的解：

$$\min_{\{\boldsymbol{a}_i\}}\sum_{i=1}^n \|\, |\boldsymbol{\Omega}_{\Lambda_0}\cdot\boldsymbol{a}_i|^2 \,\|_2^2+\lambda\left\|\boldsymbol{g}-\boldsymbol{M}\cdot[\boldsymbol{X}_1,\cdots,\boldsymbol{X}_n]\cdot\begin{pmatrix} \boldsymbol{a}_1 \\ \vdots \\ \boldsymbol{a}_n \end{pmatrix}\right\|_2^2 \tag{1.114}$$

得到解 $\{\boldsymbol{a}_i^*\}_{i=1}^n$ 后，通过贪婪迭代算法，对于第 k 迭代，计算 $\sum_i |\boldsymbol{\Omega}\cdot\boldsymbol{a}_i^*|^2$，选择 $T_k=\left\{j:\max_j \sum_i |\boldsymbol{\Omega}\cdot\boldsymbol{a}_i^*|^2, j=1,2,\cdots,p\right\}$，通过更新支撑集 $\Lambda_k=\Lambda_{k-1}-T_k$，并且计算：

$$\min_{\{\boldsymbol{a}_i\}}\sum_{i=1}^n \|\, |\boldsymbol{\Omega}_{\Lambda_k}\cdot\boldsymbol{a}_i|^2 \,\|_2^2+\lambda\left\|\boldsymbol{g}-\boldsymbol{M}\cdot[\boldsymbol{X}_1,\cdots,\boldsymbol{X}_n]\cdot\begin{pmatrix} \boldsymbol{a}_1 \\ \vdots \\ \boldsymbol{a}_n \end{pmatrix}\right\|_2^2 \tag{1.115}$$

更新得到解，直至迭代满足 $k\geqslant p-l$ 时，终止该过程，输出结果。利用这个模型可以求解如下的问题：

- 分段线性的信号或者图像的去噪；
- 分段线性的信号或者图像的分割；
- 分段线性的信号或者图像的修复。



这一部分的应用参考 2014 年 Raja Giryes、Michael Elad 和 Alfred Bruckstein 的工作。

1.2.3 稀疏认知学习、计算与识别的范式

1. 稀疏模型的最新进展

稀疏模型通常分为合成和分析两种模式，其中合成稀疏模型的研究已经比较完善，如稀疏编码、字典学习理论等；但是对于分析稀疏模型的研究相对比较“年轻”，例如该模型下的字典学习理论。对于合成稀疏模型，介绍一种结构稀疏模型，这一部分的理论及应用参考 2009 年 R. Jenatton、F. Bach 和 J. Y. Audibert 的工作和 2009 年 Junzhou Huang 和 Tong Zhang 的工作。另外对于分析稀疏模型，给出其中一种稀疏对偶框架(字典)学习的理论，这部分是 ShiDong Li 教授 2013 年的工作。

结构稀疏模型——主要给出具有结构稀疏的模型，此处的结构是指信号在字典下的表示系数及支撑集的拓扑结构。如之前考虑的合成稀疏模型中：

$$\min_{\boldsymbol{\alpha}} \|\boldsymbol{\alpha}\|_0 \quad \text{s.t.} \quad \|\boldsymbol{y}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2 \leqslant \varepsilon \tag{1.116}$$

上述模型并没有考虑$\boldsymbol{\alpha}$的支撑集 T 的结构特性。常用的一种求解该问题的方法是凸松弛算法，利用 L_1 范数代替 L_0，但是可以看到 L_1 范数是基数层面上的稀疏，编码具有较少的信息。基于此缺点，一些学者提出了比较流行的 L_1-L_2 范数，描述如下，已知$\boldsymbol{\alpha}$的指标集为 $I=\{1,2,\cdots,p\}$，给出 I 的一个有限剖分，记为 $\mathfrak{I}$，那么$\boldsymbol{\alpha}$的 L_1-L_2 范数定义为：

$$\|\boldsymbol{\alpha}\|_{L_1\text{-}L_2} = \sum_{G\in\mathfrak{I}}\left(\sum_{j\in G}\boldsymbol{\alpha}_j^2\right)^{\frac{1}{2}} \tag{1.117}$$

可以看出 L_1-L_2 范数是群集水平上的稀疏，在群内的 L_2 范数不能保证稀疏特性。但是，如果想利用结构先验信息，那么需要知道指标集给出的有限剖分 $\mathfrak{I}$ 是什么？为了回答该问题，2009 年 R. Jenatton、F. Bach 和 J. Y. Audibert 三人首先直观描述了$\boldsymbol{\alpha}$的零模式与非零模式，即

$$Z=\{j:\boldsymbol{\alpha}_j=\boldsymbol{0}, j\in I\}=\bigcup_{G\in\mathfrak{I}^S} G \tag{1.118}$$

其中 $\mathfrak{I}^S$ 为 $\mathfrak{I}$ 的一个子集，对应也给出了$\boldsymbol{\alpha}$的非零模式，即

$$P=\{j:\boldsymbol{\alpha}_j\neq\boldsymbol{0}, j\in I\}=\{G^C:G\in\mathfrak{I}\} \tag{1.119}$$

其中 G^C 为 G 的关于指标集 I 的补集。然后，研究了这两种模式与 $\mathfrak{I}$ 之间的关系，得到了两种算法，一个是已知非零模式 P，导出 $\mathfrak{I}$ 的后向算法；另外一个是已知 $\mathfrak{I}$，导出非零模式 P 和零模式 Z 的前向算法。通常关注的是后向算法，如果知道表示系数$\boldsymbol{\alpha}$的非零模式 P 或者零模式 Z，那么就可以得到 $\mathfrak{I}$，进而可以推出$\boldsymbol{\alpha}$的 L_1-L_2 范数。

另外对于贪婪算法，2009 年 Junzhou Huang 和 Tong Zhang 基于信息论编码法则提出了一种非凸惩罚，同样已知$\boldsymbol{\alpha}$的指标集为 $I=\{1,2,\cdots,p\}$，考虑一个稀疏子集 $F\subset I$，在此基础上定义了一个编码复杂度为：

$$c(F)=|F|+\mathrm{cl}(F) \tag{1.120}$$

其中 cl(F)为定义在 F 上的码长，$|F|$为 F 的基数。然后，利用$\boldsymbol{\alpha}$ 的支撑集 $\mathrm{supp}(\boldsymbol{\alpha})=\{j:\boldsymbol{\alpha}_j\neq\mathbf{0},j\in I\}$，来定义$\boldsymbol{\alpha}$ 的编码复杂度：

$$c(\alpha)=\min_F\{c(F):\mathrm{supp}(\boldsymbol{\alpha})\subset F,F\subset I\} \tag{1.121}$$

再利用$\boldsymbol{\alpha}$ 的编码复杂度作为正则项的约束，求解如下的问题：

$$\min_{\boldsymbol{\alpha}} c(\boldsymbol{\alpha}) \quad \text{s. t.} \quad \|\boldsymbol{y}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2\leqslant\varepsilon \tag{1.122}$$

或者

$$\min_{\boldsymbol{\alpha}}\|\boldsymbol{y}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2 \quad \text{s. t.} \quad c(\boldsymbol{\alpha})\leqslant s \tag{1.123}$$

其中 s 为编码复杂度。文中给出的表示系数$\boldsymbol{\alpha}$ 的结构稀疏包括：标准稀疏、群稀疏、层次稀疏、图稀疏和随机场稀疏的编码复杂度的公式。并且给出了上面问题的一种贪婪求解算法，即结构正交匹配追踪算法(Struct OMP)。最后通过实验分析得到结论：信号所对应的表示系数的编码复杂度越小，则利用编码复杂度作为正则约束得出上述问题的解的性能越好，并且能够反映出解的支撑集的结构特性。

为了讲解最优对偶框架模型，先介绍一些基础知识，包括框架的定义、对偶框架的计算公式和稀疏对偶存在的命题。

框架定义：希尔伯特空间 H 中的一组序列$\{\boldsymbol{x}_n\}_n$，如果存在 $0<A\leqslant B<+\infty$满足式子：

$$A\|\boldsymbol{f}\|_2^2\leqslant\sum_n\|<\boldsymbol{x}_n,\boldsymbol{f}>\|_2^2\leqslant B\|\boldsymbol{f}\|_2^2 \tag{1.124}$$

则称这组序列$\{\boldsymbol{x}_n\}_n$ 为 H 的一个框架，其中 A、B 分别为框架的下界和上界。当 $A=B$ 时，则该框架为紧框架；且 $A=B=1$ 时，则为帕塞瓦尔框架；如果任意去掉序列$\{\boldsymbol{x}_n\}_n$ 中的一个元素，此时不再是一个框架，该框架就称为准确框架，准确框架是一个基。

对偶框架的计算公式：对于任意一个有限非准确框架 $\boldsymbol{D}$，该框架的所有对偶框架计算具有如下的形式：

$$\widetilde{\boldsymbol{D}}=(\boldsymbol{D}\cdot\boldsymbol{D}^*)^{-1}\boldsymbol{D}+\boldsymbol{W}(\boldsymbol{I}-\boldsymbol{D}(\boldsymbol{D}^*\cdot\boldsymbol{D})^{-1}\boldsymbol{D}^*) \tag{1.125}$$

其中$\boldsymbol{W}$ 是任意的矩阵，$\boldsymbol{I}$ 是单位阵，注意此处 $\boldsymbol{D}$ 可以是稀疏表示中的过完备的字典，$\boldsymbol{D}^*$ 为 $\boldsymbol{D}$ 的希尔伯特共轭转置。

注：可以看到任意一个非准确框架 $\boldsymbol{D}$ 的对偶都有无穷多个，特别是当 $\boldsymbol{D}$ 为准确的框架时，它的对偶只有一个，即它的逆。

稀疏对偶存在：假设 $\boldsymbol{D}$ 是一个有限非准确框架(即不是基，因为基是一种特殊的框架)，以及信号 $\boldsymbol{f}$ 在 $\boldsymbol{D}$ 下是稀疏的，即 $\boldsymbol{f}=\boldsymbol{D}\cdot\boldsymbol{\alpha}$，那么存在 $\boldsymbol{D}$ 的稀疏对偶框架，记为$\widetilde{\boldsymbol{D}}$，使得$\widetilde{\boldsymbol{D}}^*\cdot\boldsymbol{f}=\boldsymbol{\alpha}$。注意对于 $\boldsymbol{D}$，$\widetilde{\boldsymbol{D}}$的选择并不是唯一的。

基于上述基础知识，考虑分析稀疏模型中的分析算子的最优选择策略，考虑的问题为：

$$\hat{\boldsymbol{f}}=\arg\min_{\boldsymbol{f}}\|\widetilde{\boldsymbol{D}}^*\cdot\boldsymbol{f}\|_0 \quad \text{s. t.} \quad \|\boldsymbol{g}-\boldsymbol{M}\cdot\boldsymbol{f}\|_2\leqslant\varepsilon \tag{1.126}$$

或者为松弛后的问题：

$$\hat{\boldsymbol{f}}=\arg\min_{\boldsymbol{f}}\|\widetilde{\boldsymbol{D}}^*\cdot\boldsymbol{f}\|_1 \quad \text{s. t.} \quad \|\boldsymbol{g}-\boldsymbol{M}\cdot\boldsymbol{f}\|_2\leqslant\varepsilon \tag{1.127}$$

其中$\widetilde{\boldsymbol{D}}$为分析算子，$\boldsymbol{M}$ 为测量矩阵。当然信号 $\boldsymbol{f}$ 不同，关于 $\boldsymbol{D}$ 的对偶$\widetilde{\boldsymbol{D}}$选取也不同。下面考虑分析算子$\widetilde{\boldsymbol{D}}$的最佳选取策略，求解问题为：

$$\{\widetilde{\boldsymbol{D}}_0, \bar{\boldsymbol{f}}\} = \arg\min_{\widetilde{\boldsymbol{D}}, \boldsymbol{f}} \| \widetilde{\boldsymbol{D}}^* \cdot \boldsymbol{f} \|_1 \quad \text{s. t.} \quad \| \boldsymbol{g} - \boldsymbol{M} \cdot \boldsymbol{f} \|_2 \leqslant \varepsilon, \boldsymbol{D} \cdot \widetilde{\boldsymbol{D}}^* = \boldsymbol{I} \tag{1.128}$$

从这一点不难看出，上式与下面的合成稀疏模型等价：

$$\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_1 \quad \text{s. t.} \quad \| \boldsymbol{g} - \boldsymbol{M} \cdot \boldsymbol{D} \cdot \boldsymbol{\alpha} \|_2 \leqslant \varepsilon \tag{1.129}$$

因此任意一个合成稀疏模型都有一个等价的分析模型。之前在合成稀疏模型中会遇到一个问题，即为什么表示系数与真正的表示系数之间差异很大，但是恢复出来的信号却能很好地接近于真实信号？基于上面的等价模型，Shidong Li 等人通过如下的定理给出了合理的解释。

定理 1.1 合成稀疏模型对应等价的分析稀疏模型的解为$\{\widetilde{\boldsymbol{D}}_0, \bar{\boldsymbol{f}}\}$，则在观测矩阵 $\boldsymbol{M}$ 满足 D-RIP 的条件下，有：

$$\| \bar{\boldsymbol{f}} - \boldsymbol{f} \|_2^2 \leqslant C_1 \varepsilon + C_2 \frac{\| \widetilde{\boldsymbol{D}}_0^* \cdot \boldsymbol{f} - (\widetilde{\boldsymbol{D}}_0^* \cdot \boldsymbol{f})_s \|_1}{\sqrt{s}} \tag{1.130}$$

其中$(\widetilde{\boldsymbol{D}}_0^* \cdot \boldsymbol{f})_s$ 为稀疏表示系数$\boldsymbol{\alpha}$ 的最佳 S 项逼近。

所以上述问题的原因是：恢复出来的信号能够良好地接近于真实信号是因为稀疏表示系数具有较快的衰减特性。

其次，Shidong Li 等人给出了另一种最优对偶选择的方法——稀疏对偶框架。考虑通过迭代的方式求解下面的问题：对于 $k=0$

$$\boldsymbol{f}_0 = \boldsymbol{D} \cdot \arg\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_0 \quad \text{s. t.} \quad \| \boldsymbol{f} - \boldsymbol{D} \cdot \boldsymbol{\alpha} \| \leqslant 0 \tag{1.131}$$

其中 $\boldsymbol{f}_0$ 是对 $\boldsymbol{f}$ 的一个粗逼近。对于 $k=1,2,\cdots$，通过求解：

$$\boldsymbol{\alpha}_{k-1} = \arg\min_{\boldsymbol{\alpha}} \| \boldsymbol{\alpha} \|_0 \quad \text{s. t.} \quad \boldsymbol{D} \cdot \boldsymbol{\alpha} = \boldsymbol{f}_{k-1} \tag{1.132}$$

然后通过对偶框架的计算公式计算：

$$\Delta\boldsymbol{\alpha}_{k-1} = \boldsymbol{\alpha}_{k-1} - \boldsymbol{D}^* (\boldsymbol{D} \cdot \boldsymbol{D}^*)^{-1} \boldsymbol{f}_{k-1} \tag{1.133}$$

其中 $\Delta\boldsymbol{\alpha}_{k-1} = (\boldsymbol{W}(\boldsymbol{I} - \boldsymbol{D}^* (\boldsymbol{D} \cdot \boldsymbol{D}^*)^{-1} \boldsymbol{D}))^* \boldsymbol{f}_{k-1}$。最后再通过下面的公式求解更新：

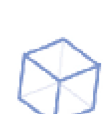
$$\boldsymbol{f}_k = \arg\min_{\boldsymbol{f}} \| \boldsymbol{D}^* (\boldsymbol{D} \cdot \boldsymbol{D}^*)^{-1} \boldsymbol{f} + \Delta\boldsymbol{\alpha}_{k-1} \|_0 \quad \text{s. t.} \quad \| \boldsymbol{g} - \boldsymbol{M} \cdot \boldsymbol{f} \|_2 \leqslant \varepsilon \tag{1.134}$$

停止迭代的条件为 $\| \boldsymbol{f}_k - \boldsymbol{f}_{k-1} \|_2 \leqslant \varepsilon$。

这种思路使用的是框架 $\boldsymbol{D}$ 的对偶框架，结合稀疏对偶存在的结论，不断地迭代以更新实现对真实信号的逼近。

2. 认知神经科学

认知神经科学是一门旨在探讨认知历程的生物学基础科学，主要的目标为阐明心理历程的神经机制，也就是大脑如何运作、如何造就心理或认知功能。它的研究主题包括注意、



意识、决策判断、学习和记忆。下面来简要地描述这些概念。

注意：是一个心理学的概念，属于认知的一部分，是一种导致局部刺激的、意识水平提高的、知觉的、选择性的集中，它表现为对某对象的指向或集中。

意识：是一个不完整、模糊的概念。一般认为意识是人对环境及自我的认知能力的清晰程度。

决策判断：指做出决定或者选择，是一种在各种替代方案中考虑各项因素做出选择的认知、思考过程。决策者在做决策之前，往往面临不同的方案和选择以及有关决定后果的某种程度上的不确定性，决策者需要对各种选择的利弊、风险做出权衡，以期达到最优的决策结果。

学习：是通过教授或者体验获得知识、技术、态度和价值的过程，学习必须依赖经验才可以有长远成效。

记忆：是指神经系统存储过往经验的能力，广泛接受的模型将记忆过程分为三个不同的阶段、编码、存储和检索。目前，认为人类的记忆过程和电脑处理信息存储的过程相似。

视觉稀疏认知进展——1968 年 Hubel 和 Wiesel 以及 1982 年 DeValois、Albrecht 和 Thorell 的工作说明，哺乳动物初级视觉皮层上的神经元的接受域(即 V1 区)具有局部、方向和频率的性质，进一步，大多数细胞被分为简单和复杂两类。在视觉研究中，基本问题就是确定为什么细胞的选择和组织具有这些性质。为此，一些学者已经考虑视觉系统与自然图像的统计性质之间的联系，给出了合理的假设，视觉系统自适应地去处理特定的输入，这种适应机制能够通过神经元的发展和演化产生。视觉输入具有特定的统计性质，如 1994 年 Ruderman 和 Bialek 给出了视觉输入不是白噪声的性质；1994 年 Field 给出了视觉输入不是高斯噪声的性质；另外，在较高层次上的描述，视觉输入包含着边缘，不同的纹理等结构性质。同时 Field 给出 V1 区的性质能够反映视觉输入的统计性质，没有一个统计信号处理系统能够处理任意类型的输入，并且达到最优，因此对于给定具有统计特性的输入集，总能够找到一个给定意义下的最优信号处理系统。

1996 年 Olshausen 和 Field 给出了在较低的层次上，简单细胞对于输入的响应可以通过一个线性模型进行描述，即

$$I(x,y)=\sum_{i=1}^{M}a_i(x,y)\cdot s_i \tag{1.135}$$

其中 $I(x,y)$为输入的图像，$a_i(x,y)$接近于相应的接受域，s_i 为简单细胞的响应，另外在这个线性模型中，Field 给出了一个基本的假设，s_i 是稀疏的。为了衡量随机变量 s_i 的稀疏性，定义期望函数 $E\{G(s_i^2)\}$，特别是 G 的选取是凸的且二阶导数是正的，如 $G(s_i^2)=s_i^4$，凸的性质暗含了在 s_i 是稀疏的时候，它的取值要么非常大，要么接近于零。另外 s_i 是统计无关的，即 s_j 不能用于预测 $s_i(i\neq j)$得到模型，求解 s_i 的过程就是稀疏编码，或者是独立成分分析(ICA)。对于独立成分分析，前提是给定大量的输入图像 $I(x,y)$，需要确定 s_i 和 $a_i(x,y)$的值，利用极大化似然的方法估计 s_i 和 $a_i(x,y)$的值。思路如下：首先考虑 $a_i(x,y)$可以形成



一个逆的线性系统，记 w_i 为 $a_i(x,y)$对应的逆滤波器。那么通过点积确定：

$$s_i = \langle w_i, I(x,y) \rangle = \sum_{x,y} w_i(x,y) \cdot I(x,y) \tag{1.136}$$

其次，假定给了 T 幅观测图像，那么似然函数可以写为：

$$\begin{aligned}\log L(I_1, I_2, \cdots, I_T, w_1, w_2, \cdots, w_M) &= \sum_{t=1}^{T}\sum_{i=1}^{M} G(\langle w_i, I_t \rangle^2) \\ &= \sum_{t=1}^{T}\sum_{i=1}^{M} G(s_{i,t}^2)\end{aligned} \tag{1.137}$$

由于 $s_{i,t}$的稀疏性，所以这里的 G 选为凸函数，那么极大化似然函数等价于极大化稀疏，最后得到的基向量 $a_i(x,y)$具有简单细胞接受域的主要性质。

2001 年 Aapo Hyvarinen 和 Patrik O. Hoyer 将这种稀疏编码的原理延拓到复杂细胞特性和结构，这里的结构指的是基函数(或细胞)的聚类组织特性。思路如下：在 ICA 的基础上，由于给出的成分是不完全相关的，能够对剩下的相关性进一步的分析。通过引入一个复杂细胞层，使得原来的稀疏编码模型变为图 1.2 的描述。因此，替代简单细胞响应 $s_{i,t}$的

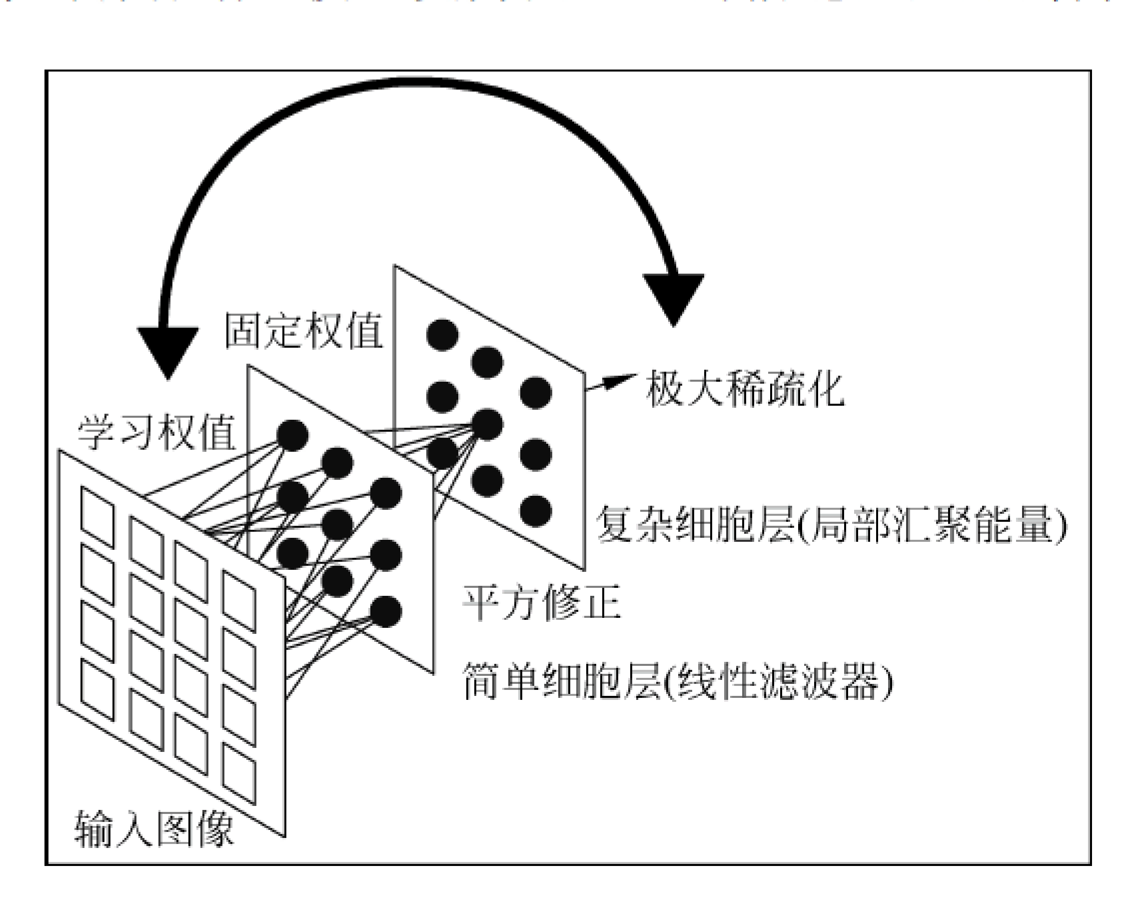


图 1.2　简单-复杂视觉皮层细胞处理图像的框图

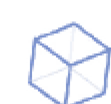
稀疏性，该模型给出了局部刺激响应的稀疏性，其中局部刺激定义为：

$$c_{i,t} = \sum_{j=1}^{M} h(i,j) \cdot \langle w_j, I_t \rangle \tag{1.138}$$

其中 $h(i,j)$为第 i 个简单细胞与第 j 个复杂细胞之间的权重函数，该权重函数不需要从输入的自然图像中学习，而是固定的。因此需要学习的仍是逆滤波器 w_j，相应的似然函数通过下式给出：

$$\log L(I_1, I_2, \cdots, I_T, w_1, w_2, \cdots, w_M) = \sum_{t=1}^{T}\sum_{i=1}^{M} G(c_{i,t}) \tag{1.139}$$

由于 $h(i,j)$是固定的，所以上面的似然函数只是关于 w_i 的函数，其中 G 是凸的，然后通过



极大似然的方法求解得到逆滤波器 w_i，进而得到 a_i。模型中的局部刺激的稀疏性指的是：在任何给定的时间，简单细胞的非零响应具有空域聚类的特性。

2007 年 Thomas Serre、Lior Wolf，Stanley Bileschi、Maximilian Riesenhuber 和 Tomaso Poggio 等人提出了类似灵长类动物视觉皮层信息处理机制的鲁棒目标识别模型。该模型处理图像的机制与灵长类动物视觉皮层中腹侧视觉通路处理自然场景的机制的关系可以通过图 1.3 来表示。

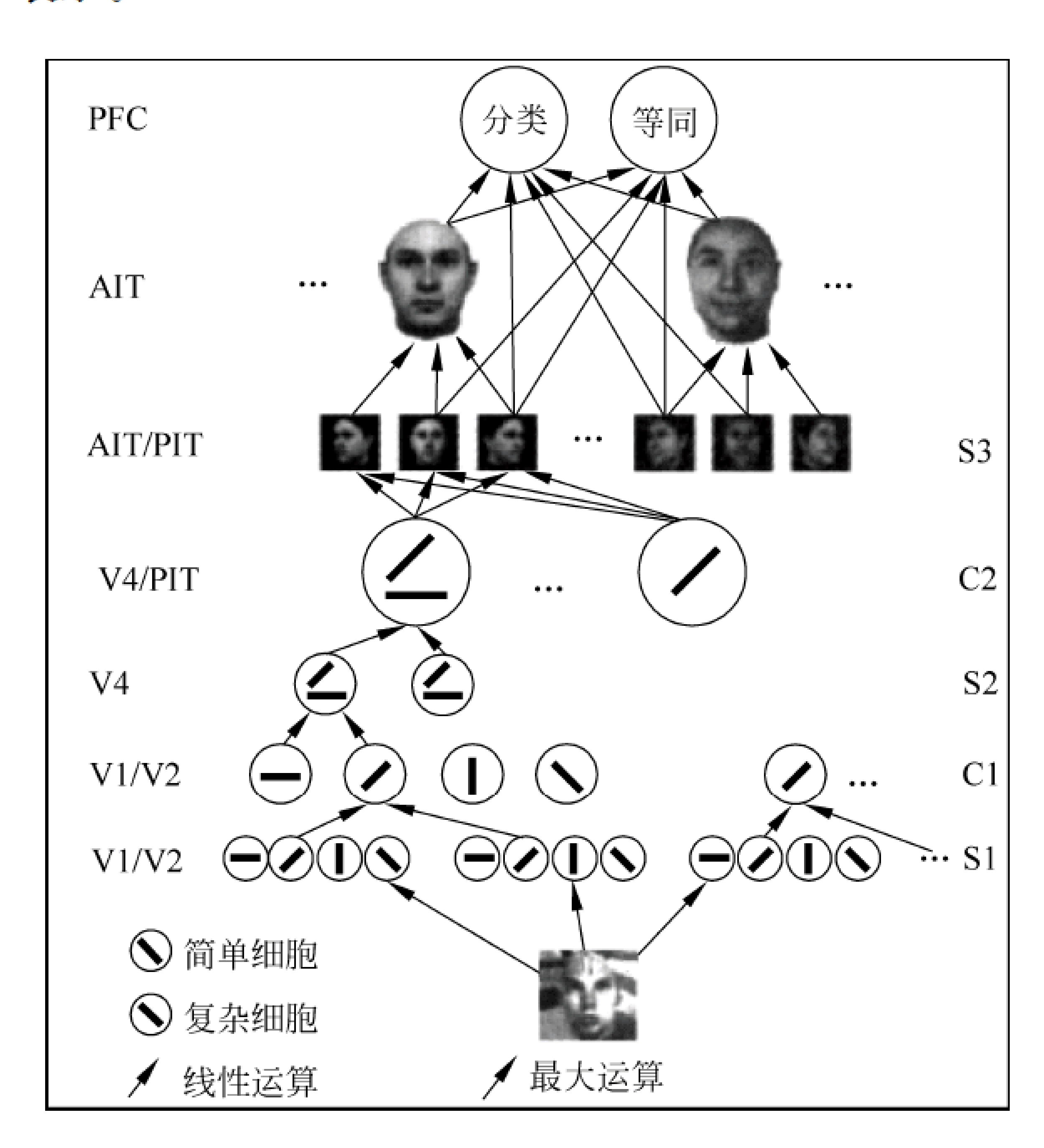
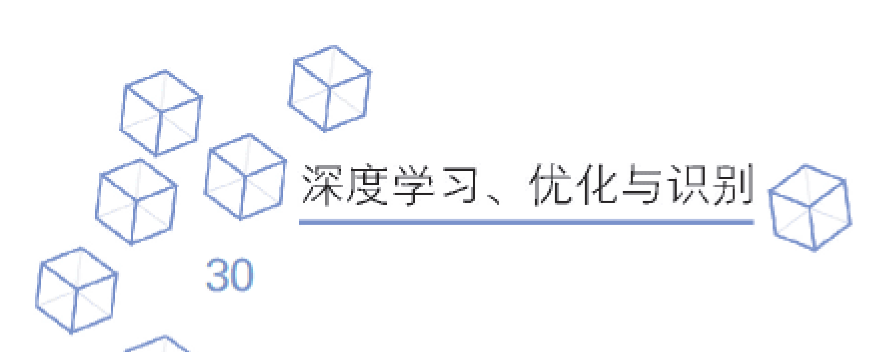


图 1.3　灵长类动物视觉皮层的等级模型和 HMAX 模型的对应关系

图 1.3 的左侧对应着视觉皮层腹侧通道中处理自然场景的等级模型，右侧对应着 HMAX 模型中的层次结构。该模型处理自然图像的主要流程可以通过图 1.4 进行描述。该模型主要基于 Maximilian Riesenhuber 和 Tomaso Poggio 两人提出的层次目标识别算法(即 HMAX 模型)，通过简单细胞单元 S 与复杂细胞单元 C 的交替组成，S 单元采用 TUNING 操作，用于增加目标的选择特性；C 单元具有更大的感受野范围，它通过 MAX 操作汇聚 S 单元的输出，从而引入了对目标尺度和平移的不变性。在此基础上，通过在该模型的第三层 S2 上引入特征编码字典，利用字典来表征 C1 层上的输出响应，得到 S2 层上的响应，再通过局部极大值操作实现最后一层 C2 层上的中层特征提取；最后通过简单的分类器设计，实现复杂场景下的目标识别。下面详细地叙述该模型每一层上的操作，以及它所对应的神经生理方面的解释。

S1 层：对于一幅灰度图像通过 S1 层上的分析，对应 Hubel 和 Wiesel 在哺乳类动物的



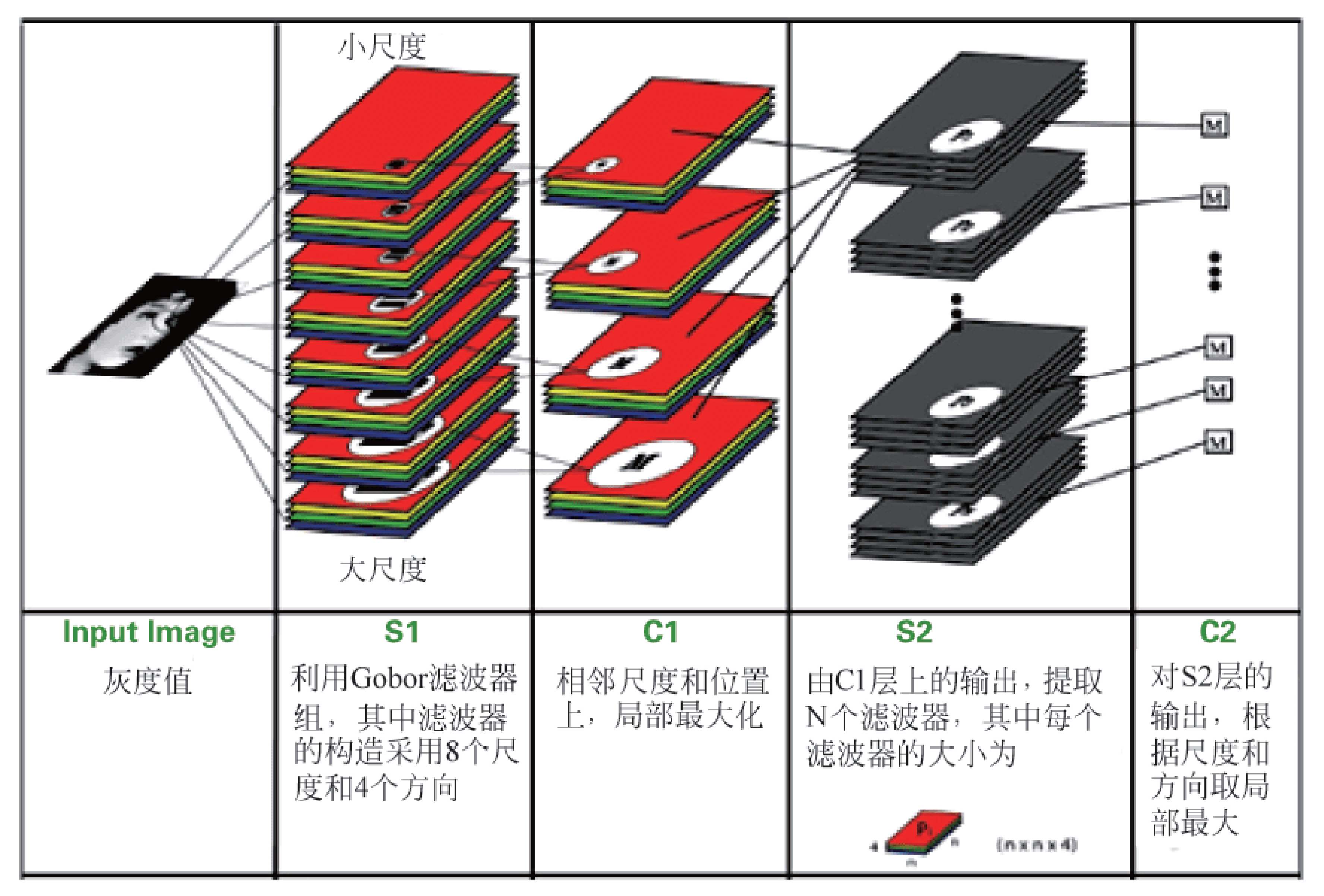


图 1.4　HMAX 模型

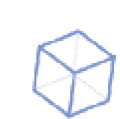
初级视觉皮层(即 V1 区)简单细胞的响应。S1 层上主要取已经能够较好地描述初级皮层上简单细胞的感受野的 Gabor 函数形式，即

$$F(x,y)=\mathrm{e}^{-\frac{(x_0^2+r^2y_0^2)}{2\sigma^2}}\cos\left(\frac{2\pi}{\lambda}x_0\right)$$

$$\begin{cases}x_0=x\cos\theta+y\sin\theta\\ y_0=-x\sin\theta+y\cos\theta\end{cases}\tag{1.140}$$

通过 σ 和 λ 控制滤波器的尺寸(文中取从 7×7 到 37×37，每隔 2 个单位，共 16 个尺度)，用 θ 来控制方向 $\left(\theta=0,\frac{\pi}{4},\frac{\pi}{2},\frac{3\pi}{4}\right)$，一共产生 64 个滤波器，对每一幅图像用这 64 个滤波器进行滤波，得到 64 幅图。

C1 层：该层主要对应着初级视觉皮层上的复杂细胞的平移和尺寸的不变特性，并且复杂细胞相对于简单细胞，具有较大感受野。C1 层主要对 S1 层上的响应做局部极大值和汇聚处理，即对每相邻两个尺度，构成一个带，这样便可以形成 8 个带，如带 1 就是 7×7 和 9×9 尺度上所对应的滤波器的输出(每一个尺度上都是 4 个方向)；在每一个带上，有两组图像，每一组都是特定的滤波器下的 4 个不同方向上所形成的 4 幅图，并对每个带进行如下的操作，两组中对应着相同方向的 2 幅图像进行滑块处理，每一幅图像先在给定的窗口尺寸及重叠度的设置下，进行滑窗处理，每一个窗口内的最大值记为该窗口的代表点，所有这些代表点组成这幅图像的新描述；然后对这两幅图的新描述作比较，大的保留小的去掉，形成



该带上特定方向的描述图，此带上有 4 个方向，故有 4 幅描述。共计 8 带，所以有相应的 32 幅描述，以此作为 C1 层上的响应输出。

S2 层：在 C1 层中的响应上进行简单的随机采样或者学习的方式，抽取或学习得到 N"块"，每一"块"上有 4 个方向，大小为 $n\times n\times 4$，其中 $n=4,8,12,16$；每一"块"与 C1 层上的响应进行匹配计算，当"块"与 C1 中的每一带上的大小不一样时，可以对"块"或带进行插值或者抽样，它的匹配度通过如下的公式来计算：

$$r_i = \mathrm{e}^{-\beta\|\boldsymbol{X}-\boldsymbol{P}_i\|^2} \tag{1.141}$$

其中 $\boldsymbol{X}$ 为 C1 层上的响应，$i=1,2,\cdots,N$，对于每一"块"，与 C1 层上的每一个带上进行匹配，可以得到 1×8 的一个行向量。共计有 N 块，可以得到 $N\times 8$ 的矩阵。

C2 层：对 S2 层得到的矩阵，对每一行取最大值，最终得到一个 $N\times 1$ 的特征向量作为这幅图像的描述。C2 层上的响应，具有平移和位置的不变性表征。

最后，在学习得到的特征向量的基础上，通过简单的分类器设计(SVM 或者 Boosting)来实现复杂场景下的目标识别。SVM 应用于多类判别的问题可以采取两种方式，一种是一对多方法；另一种是一对一方法。目前，关于视觉皮层的生理研究，已经处于层次目标识别阶段，并且从自底向上和自顶向下的双向等级模型中解释复杂场景下的目标识别机制。对应着这些生理研究方面所取得的进展，稀疏神经编码的计算模型也从早期的稀疏编码到结构稀疏、判别稀疏模型，发展到现在的前馈式信息传递的层次化稀疏模型。稀疏神经认知是一种基于哺乳动物视觉皮层信息处理的机制，其为神经生理基础的广义目标识别计算模型。与之前的稀疏编码不太一样的地方是：该计算模型可以处理更为复杂场景下的目标识别问题，并且模仿哺乳动物 V4 区和颞下皮层上细胞感受野的特性，使其具有变换不变性的中层特征，这些特征在目标的判断与识别任务中，较之前的初级视觉皮层上(稀疏编码)所获取得到的边缘、纹理、轮廓等初级特征具有一定的优势。

1.3 机器学习与神经网络

机器学习与深度学习的根本性差别是追求的理念不同。机器学习在中小规模的数据上追求精度与效率，所以花费大量的时间研究数据的先验特性，并把它加到特征学习中学习得到分布式判别性特征，模型可以拆分为特征学习和分类器/回归器设计；而深度学习的理念是在中大规模数据集上追求简单、新颖和通用，放弃精确，不强调数据的先验特性，模型讲究统一的端到端的设计方式。下面介绍深度学习中仍沿(使)用的机器学习和神经网络的框架或理论基础。

1.3.1 机器学习

众所周知，数据是载体，智能是目标，而机器学习是从数据通往智能的技术与途径；因此，机器学习是数据科学的核心，是现代人工智能的本质，它可以从数据中挖掘出有价值、有

规律的信息，其通用的框架为数据、模型、优化和求解。下面从框架结构出发，主要陈述支撑矢量积、贝叶斯分类器和强化学习这三个方面。

1. 支撑向量机

问题：给定训练样本集，其中输入为向量、输出为类标集；如何基于该数据集在样本空间找到一个超平面，将不同类别的样本分开。若输入为矩阵，要求在不进行向量化操作的前提下进一步改进算法（支撑矩阵机、支撑张量机）。

数据：训练数据集为$\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$，其中$\boldsymbol{x}_i \in \mathbb{R}^n$，$y_i \in \{-1, +1\}$。

模型：假设数据集是线性可分的，则利用线性模型，即

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b \tag{1.142}$$

其中$\boldsymbol{w}$和b是模型参数。

优化：优化目标函数为：

$$\begin{aligned} &\min_{\boldsymbol{w}, b} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ &\text{s.t.} \quad y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b) \geqslant 1, \quad i = 1, 2, \cdots, N \end{aligned} \tag{1.143}$$

求解：优化目标函数为一个凸的二次规划问题，可以直接利用现成的优化计算方法求解。另外，根据 KKT 条件，也可以利用其对偶问题求解，首先，原问题为：

$$\max_{\boldsymbol{\alpha} \geqslant 0} \min_{\boldsymbol{w}, b} L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \sum_i \alpha_i(1 - y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b)) \tag{1.144}$$

其中$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_N)$，其对偶问题为：

$$\min_{\boldsymbol{w}, b} \max_{\boldsymbol{\alpha} \geqslant 0} L(\boldsymbol{w}, b, \boldsymbol{\alpha}) \tag{1.145}$$

通过固定$\boldsymbol{w}$和b，求解$\boldsymbol{\alpha}$，即目标函数$L(\boldsymbol{w}, b, \boldsymbol{\alpha})$关于$\boldsymbol{w}$和$b$的偏导数为零得到如下的条件$\boldsymbol{w} = \sum_i \alpha_i \cdot y_i \cdot \boldsymbol{x}_i$和$\sum_i \alpha_i \cdot y_i = 0$，代入$L(\boldsymbol{w}, b, \boldsymbol{\alpha})$可以消去$\boldsymbol{w}$和$b$，仅得到关于$\boldsymbol{\alpha}$的目标函数：

$$\begin{aligned} &\max_{\alpha \geqslant 0} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \cdot \alpha_j \cdot y_i \cdot y_j \cdot \boldsymbol{x}_i^{\mathrm{T}} \cdot \boldsymbol{x}_j \\ &\text{s.t.} \quad \sum_i \alpha_i \cdot y_i = 0 \end{aligned} \tag{1.146}$$

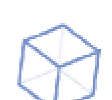
求出$\boldsymbol{\alpha}$后便可以得到$\boldsymbol{w}$；其中偏置b的求解通过平均所有支撑向量获取：

$$y_s \cdot \left(\sum_{s \in S} \alpha_i \cdot y_i \cdot \boldsymbol{x}_i^{\mathrm{T}} \cdot \boldsymbol{x}_s + b\right) = y_s \cdot \left(\sum_{s \in S} \boldsymbol{w}^{\mathrm{T}} \cdot \boldsymbol{x}_s + b\right) = 1 \tag{1.147}$$

其中$S = \{i : \alpha_i > 0, i = 1, 2, \cdots, N\}$为支撑向量的指标集。

改进一：从问题出发，通过数据、模型、优化和求解四部分，分析了数据线性可分情况下的支撑向量机。但是，如果线性不可分时，需要通过特征学习的方式对数据进行处理（例如升维等），使得处理后的数据是线性可分的，假设通过如下的投影方式进行特征学习：

$$X = \varphi(\boldsymbol{x}) \tag{1.148}$$



那么上面的优化目标，便写为：

$$\begin{aligned}&\min_{\boldsymbol{w},b}\frac{1}{2}\|\boldsymbol{w}\|^2\\&\text{s.t.}\quad y_i(\boldsymbol{w}^{\mathrm{T}}\varphi(\boldsymbol{x}_i)+b)\geqslant 1,\quad i=1,2,\cdots,N\end{aligned}\tag{1.149}$$

相应的对偶问题中，关于 α 的目标函数为：

$$\begin{aligned}&\max_{\boldsymbol{\alpha}\geqslant 0}\sum_i\alpha_i-\frac{1}{2}\sum_{i,j}\alpha_i\cdot\alpha_j\cdot y_i\cdot y_j\cdot\varphi(\boldsymbol{x}_i^{\mathrm{T}})\cdot\varphi(\boldsymbol{x}_j)\\&\text{s.t.}\quad\sum_i\alpha_i\cdot y_i=0\end{aligned}\tag{1.150}$$

为了避免直接运算 $\varphi(\boldsymbol{x}_i^{\mathrm{T}})\cdot\varphi(\boldsymbol{x}_j)$（因为特征空间的维数可能很高，甚至可能是无穷维），可以引入函数：

$$\kappa(\boldsymbol{x}_i,\boldsymbol{x}_j)=\varphi(\boldsymbol{x}_i^{\mathrm{T}})\cdot\varphi(\boldsymbol{x}_j)\tag{1.151}$$

这个函数 $\kappa(\cdot,\cdot)$ 就是核函数。常用的核函数有线性核、多项式核、拉普拉斯核以及高斯核等。

改进二：当输入为矩阵时，为避免向量化可能会引起数据的拓扑结构变化，而引入支撑矩阵机，其中数据为 $\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$，其中 $\boldsymbol{x}_i\in\mathbb{R}^{n\times m}$，$y_i\in\{-1,+1\}$，优化目标函数为：

$$\min_{\boldsymbol{w},\boldsymbol{b}}\frac{1}{2}\mathrm{tr}(\boldsymbol{W}^{\mathrm{T}}\cdot\boldsymbol{W})+\tau\|\boldsymbol{W}\|_*+C\sum_i h(y_i\cdot\mathrm{tr}(\boldsymbol{W}^{\mathrm{T}}\cdot\boldsymbol{x}_i+\boldsymbol{b}))\tag{1.152}$$

其中 $\|\cdot\|_*$ 为核范数，即 $\|\boldsymbol{W}\|_*$ 为矩阵 $\boldsymbol{W}$ 奇异值的和，为了保持局部结构特性，$h(\cdot)$ 为 hinge-loss 函数：

$$h(u)=\begin{cases}0 & u>1\\1-u & u\leqslant 1\end{cases}\tag{1.153}$$

求解方法为参数迭代优化。

改进三：为了缓解（假设）线性可分与线性不可分这两种极端情形，即存在着一个超平面将数据完全分开（所有样本完全划分正确可能会引起学习参数任务困难），所以引入软间隔策略，即在假设的前提下允许一些样本的划分出错，从而改进优化目标函数为：

$$\begin{aligned}&\min_{\boldsymbol{w},b}\frac{1}{2}\|w\|^2+C\sum_{i=1}^m\xi_i\\&\text{s.t.}\quad y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b)\geqslant 1-\xi_i,\\&\qquad\quad \xi_i\geqslant 0,i=1,2,\cdots,N\end{aligned}\tag{1.154}$$

这就是常用的软间隔支撑向量机，通过拉格朗日乘子法得到的增广拉格朗日函数为：

$$\begin{aligned}&\max_{\boldsymbol{\alpha}\geqslant 0,\mu\geqslant 0}\ \min_{\boldsymbol{w},b,\xi}L(\boldsymbol{w},b,\xi,\boldsymbol{\alpha},\mu)\\&=\frac{1}{2}\|\boldsymbol{w}\|^2+C\sum_{i=1}^m\xi_i+\sum_{i=1}^m\alpha_i(1-\xi_i-y_i(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i+b))-\sum_{i=1}^m\mu_i\cdot\xi_i\end{aligned}\tag{1.155}$$

求解与之前类似。

改进四：当输入数据为 $x\in\mathbb{R}^{d_1\times d_2\times\cdots\times d_N}$，为了保持数据的拓扑结构特性，引入支撑张量



机，数据集为$\{(x_i,y_i)\}_{i=1}^N$，其中$y_i\in\{-1,+1\}$，优化目标函数为：

$$\begin{aligned}&\min_{w,b}\frac{1}{2}\|\boldsymbol{w}_{d_1}\circ\boldsymbol{w}_{d_2}\circ\cdots\circ\boldsymbol{w}_{d_N}\|^2+C\sum_{i=1}^m\xi_i\\&\text{s.t.}\quad y_i\left(x_i\prod_{n=1}^N\times_n\boldsymbol{w}_{d_n}+b\right)\geqslant1-\xi_i,\xi_i\geqslant0,i=1,2,\cdots,N\end{aligned}\tag{1.156}$$

其中的两个操作解释如下：第一个是外积，即

$$W=\boldsymbol{w}_{d_1}\circ\boldsymbol{w}_{d_2}\circ\cdots\circ\boldsymbol{w}_{d_N}\in\mathbb{R}^{d_1\times d_2\times\cdots\times d_N}\tag{1.157}$$

其中$\boldsymbol{w}_{d_i}\in\mathbb{R}^{d_i}$，那么$W$中每一个元素的值通过如下的公式求解：

$$\begin{cases}W(i_1,i_2,\cdots,i_N)=\boldsymbol{w}_{d_1}(i_1)\cdot\boldsymbol{w}_{d_2}(i_2)\cdot\cdots\boldsymbol{w}_{d_N}(i_N)\\i_1=1,2,\cdots,d_1\\i_2=1,2,\cdots,d_2\\\vdots\\i_N=1,2,\cdots,d_N\end{cases}\tag{1.158}$$

第二个是d_N模态的积，即

$$\begin{aligned}x\prod_{n=1}^N\times_n\boldsymbol{w}_{d_n}&=x\times_1\boldsymbol{w}_{d_1}\times_2\boldsymbol{w}_{d_2}\times\cdots\times_N\boldsymbol{w}_{d_N}\\&=\left(\sum_{i_1=1}^{d_1}x_{i_1,}\cdot\boldsymbol{w}_{d_1}(i_1)\right)^{\xrightarrow{\text{size}}\mathbb{R}^{d_2\times d_3\times\cdots\times d_N},\text{denote }x^1}\times_2\boldsymbol{w}_{d_2}\cdots\times_N\boldsymbol{w}_{d_N}\\&=\left(\sum_{i_2=1}^{d_2}x^1_{i_2,}\cdot\boldsymbol{w}_{d_2}(i_2)\right)^{\xrightarrow{\text{size}}\mathbb{R}^{d_3\times d_4\times\cdots\times d_N},\text{denote }x^2}\times_3\boldsymbol{w}_{d_3}\cdots\times_N\boldsymbol{w}_{d_N}\\&=x^N\in\mathbb{R}\end{aligned}\tag{1.159}$$

求解与之前类似，利用拉格朗日增广目标函数的对偶问题求解。

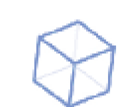
2. Softmax 分类器

Softmax 函数是将多个标量映射为一个概率分布，对于训练数据集$\{(\boldsymbol{s}_i,y_i)\}_{i=1}^N$且$\boldsymbol{s}_i\in\mathbb{R}^{n\times m}$，$y_i\in\{1,2,\cdots,C\}$，其中$C$为类别个数。通常，通过特征学习将数据集映射为$\{(\boldsymbol{x}_i,y_i)\}_{i=1}^N$，其中$\boldsymbol{x}_i\in\mathbb{R}^m$；模型为：

$$\hat{\boldsymbol{y}}=P(y\mid\boldsymbol{x})=\begin{bmatrix}P(y=1\mid\boldsymbol{x})\\P(y=2\mid\boldsymbol{x})\\\vdots\\P(y=C\mid\boldsymbol{x})\end{bmatrix}=\frac{1}{\sum\limits_{i=1}^{C}\mathrm{e}^{\boldsymbol{x}^\mathrm{T}\cdot\theta_i}}\cdot\begin{bmatrix}\mathrm{e}^{\boldsymbol{x}^\mathrm{T}\cdot\theta_1}\\\mathrm{e}^{\boldsymbol{x}^\mathrm{T}\cdot\theta_2}\\\vdots\\\mathrm{e}^{\boldsymbol{x}^\mathrm{T}\cdot\theta_C}\end{bmatrix}\in\mathbb{R}^C\tag{1.160}$$

通过交叉熵建立优化目标函数：

$$\min_\theta\frac{1}{N}\sum_{i=1}^N\left[-\sum_{c=1}^C\delta(y_i=c)\cdot\log P(y_i=c\mid\boldsymbol{x}_i,\theta)\right]+\lambda R(\theta)\tag{1.161}$$



其中 δ 为狄利克雷函数，参数$\boldsymbol{\theta}=(\theta_1,\theta_2,\cdots,\theta_C)$，$R(\cdot)$为正则化约束项。利用梯度下降的方法求解参数$\boldsymbol{\theta}$，便得到 Softmax 分类器。

3. 强化学习

强化学习是机器学习的一个分支，它讲究在一系列的情景之下，通过多步恰当的决策来达到一个目标，是一种序列多步决策的问题。与传统的机器学习算法不同，需要对情景及恰当的决策之间进行搜索，根据反馈对这种搜索策略进行奖罚，与人类与环境的交互方式类似。强化学习任务通常用马尔可夫决策过程来描述：包括环境 E、状态空间 X、机器搜索策略 P、动作空间 A、反馈机制 V(即当前环境对搜索策略选择下的动作的积极/消极反馈)等，其中每个状态 $x\in X$ 为感知器对当前环境的描述，在某个合理的策略选择下 $p\in P$(需要训练学习)，得到的动作 $a\in A$ 作用在当前的状态 x 上，使得环境从当前状态按照某种概率转移到另一个状态 $y\in X$；状态改变后的环境会根据从状态 x 到状态 y 的积极或消极特性(这种特性通过条件概率来刻画)对这个策略选择 p 进行奖罚。

策略作为状态到动作之间的映射：

$$a=p(x) \tag{1.162}$$

即在状态 x 下，根据策略 p，采取动作 a。有了这个定义，那么需要回答两个问题：一是策略如何选取；二是如何判断一个策略的好坏；这两个问题是相辅相成的，可以通过一个值函数：

$$r_t=V(x_t,a_t=p(x)) \tag{1.163}$$

即在策略 p 下，从状态 x_t 出发，执行动作 a_t 所带来的累积奖赏。

那么寻找策略的目标便是让累积的奖赏最大，这里需要强调的是：大部分强化学习的模型都是定义在马尔可夫链上的，即状态、动作、累积奖赏，再到下一个状态、动作、累积奖赏等；且下一个状态只与当前的状态有关，与以前的状态没有关系；基于此，易得到如下的一个递归式子：

$$\begin{aligned}V(x,a)&=E[r_{t+1}+\eta\cdot r_{t+2}+\eta^2\cdot r_{t+3}+\cdots\mid x_t,a_t]\\&=E[r_{t+1}+\eta\cdot V(\tilde{x},\tilde{a})\mid x_t,a_t]\end{aligned} \tag{1.164}$$

其中的 η 为折扣系数。最优的值函数可写为：

$$V^*(x,a)=E[r_{t+1}+\eta\cdot V^*(\tilde{x},\tilde{a})\mid x_t,a_t] \tag{1.165}$$

这就是著名的 Bellman 方程，它是求解值函数的关键，它可以按照马尔可夫过程展开形成一个树状的结构，通过将这些奖赏综合起来，然后便可以将得到状态、动作、累积奖赏不断地迭代下去。

但通常无法做到对所有的状态和动作进行展开(空间体量太大)，那么像种瓜一样，一次种不好，就多种几次，通过求取平均的累积奖赏来作为期望累积奖赏的近似，这样便可以估算值函数了。假设已经有 N 次实验估算出一个值函数为 $V_N(x_t,a_t)$，如果现在增加一次实验，那么可以在原来的值函数上进行更新得到新的值函数：

$$V_{N+1}(x_t,a_t)=V_N(x_t,a_t)+\alpha[r_{t+1}+\eta\cdot V_N(x_{t+1},a_{t+1})-V_N(x_t,a_t)] \tag{1.166}$$



此处 α 为学习速率(调整步长)。一旦得到值函数,就得到了最优策略,在当前的状态下,通过枚举动作集合,选择最大的值函数所对应的动作即可。

在人工智能领域,感知和决策能力是衡量智能的指标。深度学习具有较强的感知能力,但是缺乏一定的决策能力;而强化学习具有决策能力,但对感知问题束手无策;因此,将两者结合起来形成深度强化学习,优势互补,为复杂系统的感知决策问题提供了解决思路。

1.3.2 神经网络

神经网络是由具有自适应的简单单元组成的广泛、并行、互联的网络,它的组织能够模拟生物神经系统对真实世界物体所做出的交互反应。通常,在机器学习中谈论的神经网络指的是神经网络学习,或者说是机器学习与神经网络这两个学科领域的交叉部分。

1. 径向基网络

径向基网络是一种单隐层神经网络,它使用径向基函数作为隐层神经元激活函数,输出层则是隐层神经元输出的线性组合,即输入 $\boldsymbol{x}\in\mathbb{R}^n$,输出 $y\in\mathbb{R}$,模型为:

$$\begin{cases} y=\sum_{i=1}^{K}\omega_i\cdot\rho(\boldsymbol{x},c_i,\beta_i) \\ \rho(\boldsymbol{x},c_i,\beta_i)=\mathrm{e}^{-\beta_i\cdot\|x-c_i\|^2} \end{cases} \tag{1.167}$$

其中 K 为隐层节点的个数,ρ 为径向基函数,参数 c_i,β_i 分别为中心与乘性偏置。利用预测与期望输出之间的平方差最小求解参数 c_i,β_i,ω_i,得到的优化目标函数为:

$$\min_{c,\beta,\boldsymbol{\omega}}\frac{1}{N}\sum_{i=1}^{N}\left\|y_i-\sum_{j=1}^{K}\omega_j\cdot\rho(\boldsymbol{x}_i,c_j,\beta_j)\right\|_2^2+\lambda\|\boldsymbol{\omega}\|_F^2 \tag{1.168}$$

求解通常包括两步,一是确定隐层上每一个单元的中心 c_i,通过随机采样或者聚类方法等;二是利用梯度下降的方法确定参数 β_i,ω_i。

注意: Park 和 Sandberg 于 1991 年证明具有足够多的隐单元个数的径向基网络能以任意的精度逼近任意连续函数。

2. 玻尔兹曼机/受限玻尔兹曼机

玻尔兹曼机或者受限玻尔兹曼机是一种基于能量的模型,即能量最小化时网络模型达到理想状态。网络结构分为两层:显层 $\boldsymbol{v}\in\{0,1\}^n$ 用于数据的输入与输出,隐层 $\boldsymbol{h}\in\{0,1\}^m$ 则被理解为数据的内在表达。

受限玻尔兹曼机:数据集为 $\{\boldsymbol{v}_i\}_{i=1}^N$(本质上,波尔兹曼机和受限波尔兹曼机为自编码网络,是一种无监督学习方式),关于受限(同一层的单元相互不连接)波尔兹曼机建立的能量函数为:

$$E(\boldsymbol{v},\boldsymbol{h})=-(\boldsymbol{a}^{\mathrm{T}}\cdot\boldsymbol{v}+\boldsymbol{b}^{\mathrm{T}}\cdot\boldsymbol{h}+\boldsymbol{v}^{\mathrm{T}}\cdot\boldsymbol{w}\cdot\boldsymbol{h}) \tag{1.169}$$

基于能量函数,可以建立 $\boldsymbol{v},\boldsymbol{h}$ 的联合分布函数:

$$\begin{cases} P(\boldsymbol{v},\boldsymbol{h}) = \dfrac{1}{Z}\mathrm{e}^{-E(\boldsymbol{v},\boldsymbol{h})} \\ Z = \sum\limits_{\boldsymbol{v},\boldsymbol{h}} \mathrm{e}^{-E(\boldsymbol{v},\boldsymbol{h})} \end{cases} \tag{1.170}$$

对于每一个样本$\boldsymbol{v}$，在参数$\theta=(a,b,\boldsymbol{w})$确定下，通过下式便得到对$\boldsymbol{v}$的一个估计：

$$\boldsymbol{v} \xrightarrow{P(\boldsymbol{h} \mid \boldsymbol{v},\theta)} \boldsymbol{h} \xrightarrow{P(\boldsymbol{v} \mid \boldsymbol{h},\theta)} \hat{\boldsymbol{v}} \tag{1.171}$$

其中的两个条件概率分布计算如下：

$$\begin{cases} P(\boldsymbol{h} \mid \boldsymbol{v},\theta) = \dfrac{1}{\tilde{Z}_{\boldsymbol{v}}} \cdot \mathrm{e}^{(\boldsymbol{b}^{\mathrm{T}}\cdot\boldsymbol{h}+\boldsymbol{v}^{\mathrm{T}}\cdot\boldsymbol{w}\cdot\boldsymbol{h})} \\ \tilde{Z}_{\boldsymbol{v}} = P(\boldsymbol{v}) \cdot Z \end{cases} \quad \begin{cases} P(\boldsymbol{v} \mid \boldsymbol{h},\theta) = \dfrac{1}{\tilde{Z}_{\boldsymbol{h}}} \cdot \mathrm{e}^{(\boldsymbol{a}^{\mathrm{T}}\cdot\boldsymbol{v}+\boldsymbol{v}^{\mathrm{T}}\cdot\boldsymbol{w}\cdot\boldsymbol{h})} \\ \tilde{Z}_{\boldsymbol{h}} = P(\boldsymbol{h}) \cdot Z \end{cases} \tag{1.172}$$

进一步，在数据集上构建优化目标函数为：

$$\begin{aligned} \max_{\theta} J(\theta) &= \sum_{i=1}^{N} \log P(\hat{\boldsymbol{v}}_i) = \sum_{i=1}^{N} \log \sum_{h} P(\hat{\boldsymbol{v}}_i,\boldsymbol{h}) \\ &= \left(\sum_{i=1}^{N} \log \sum_{\boldsymbol{h}} \mathrm{e}^{-E(\hat{\boldsymbol{v}}_i,\boldsymbol{h})}\right) - N\log Z \\ &= \left(\sum_{i=1}^{N} \log \sum_{\boldsymbol{h}} \mathrm{e}^{-E(\hat{\boldsymbol{v}}_i,\boldsymbol{h})}\right) - N\log \sum_{\boldsymbol{v},\boldsymbol{h}} \mathrm{e}^{-E(\hat{\boldsymbol{v}},\boldsymbol{h})} \end{aligned} \tag{1.173}$$

求解通过对比散度算法，即如下的公式估计参数值：

$$\begin{cases} \dfrac{\partial J}{\partial w_{i,j}} \approx \dfrac{1}{N}\sum\limits_{n=1}^{N} \boldsymbol{v}_n(i)\boldsymbol{h}_n(j) - \dfrac{1}{N}\sum\limits_{n=1}^{N} \hat{\boldsymbol{v}}_n(i)\,\hat{\boldsymbol{h}}_n(j) \\ \dfrac{\partial J}{\partial a_i} \approx \dfrac{1}{N}\sum\limits_{n=1}^{N} \boldsymbol{v}_n(i) - \dfrac{1}{N}\sum\limits_{n=1}^{N} \hat{\boldsymbol{v}}_n(i) \\ \dfrac{\partial J}{\partial b_j} \approx \dfrac{1}{N}\sum\limits_{n=1}^{N} \boldsymbol{h}_n(i) - \dfrac{1}{N}\sum\limits_{n=1}^{N} \hat{\boldsymbol{h}}_n(i) \end{cases} \tag{1.174}$$

其中$\hat{\boldsymbol{v}}$，$\hat{\boldsymbol{h}}$为估计值，即通过：

$$\boldsymbol{v} \longrightarrow \boldsymbol{h} \longrightarrow \hat{\boldsymbol{v}}_1 \longrightarrow \hat{\boldsymbol{h}}_1 \longrightarrow \hat{\boldsymbol{v}}_2 \longrightarrow \hat{\boldsymbol{h}}_2 \longrightarrow \cdots \tag{1.175}$$

若利用$(\hat{\boldsymbol{v}},\hat{\boldsymbol{h}})=(\hat{\boldsymbol{v}}_1,\hat{\boldsymbol{h}}_1)$代入估算式子中，则称为一阶对比散度算法；若使用$(\hat{\boldsymbol{v}},\hat{\boldsymbol{h}})=(\hat{\boldsymbol{v}}_2,\hat{\boldsymbol{h}}_2)$，则称为二阶对比散度算法，以此类推，得到$k$阶对比散度算法。

玻尔兹曼机：与受限波尔兹曼机的区别在于：不限制同一层的单元是相互独立的，即可以相互连接，其推导公式与受限玻尔兹曼机类似，记$\boldsymbol{x}=(\boldsymbol{x}_v,\boldsymbol{x}_h)\in\mathbb{R}^{n+m}$且$\boldsymbol{x}_v=\boldsymbol{v}$，$\boldsymbol{x}_h=\boldsymbol{h}$，其能量函数为：

$$E(\boldsymbol{x}) = -(\boldsymbol{b}^{\mathrm{T}} \cdot \boldsymbol{x} + \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{w} \cdot \boldsymbol{x}) \tag{1.176}$$

进一步，概率密度函数为：

$$P(\boldsymbol{x}) = \frac{1}{Z}\mathrm{e}^{-E(\boldsymbol{x})} \tag{1.177}$$

其中 $Z=\sum_{x} \mathrm{e}^{-E(x)}$，假设给定的数据集服从独立同分布，则优化目标函数为：

$$\max_{\theta} J(\theta)=\frac{1}{N} \sum_{n} \log P(\boldsymbol{x}_{v}^{n}) \tag{1.178}$$

其中 $\boldsymbol{x}_{v}^{n}$ 是第 n 个样本 $\boldsymbol{v}^{n}$，求解与之前的对比散度算法类似。

3. 小波神经网络

小波神经网络是指沿用神经网络的结构，其中超参数（激活函数）选择小波函数 $\psi(t)$（具有解析形式），选定特定的小波后，其参数包括尺度因子和平移因子。

训练数据集为 $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$，其中 $\boldsymbol{x}_i \in \mathbb{R}^n, y_i \in \mathbb{R}$，进一步，建立的网络模型（三层结构，即输入、隐层（节点个数为 m）和输出为：

$$\begin{cases} u_i = \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{w}(:, i) \\ h_i = \psi\left(\dfrac{u_i - b_i}{a_i}\right) \\ y = \sum\limits_{i=1}^{m} c_i \cdot h_i \end{cases} \tag{1.179}$$

这里的 u_i、h_i 分别为隐层第 i 个节点的输入与输出，其中网络参数为：

$$\theta=(w, b, a, c) \tag{1.180}$$

基于模型建立的优化目标函数为：

$$\min_{\theta} J(\theta)=\frac{1}{N} \sum_{n=1}^{N}\left\|y_i-\hat{y}_i\right\|_2^2+\lambda R(\theta) \tag{1.181}$$

其中 $\hat{y}_i$ 为预测输出，$R(\theta)$ 为正则项（约束在权值矩阵 $\boldsymbol{w}$ 和 c 上）。求解依旧采用梯度下降的方式。

神经生理学家的最新研究表明：大脑的视觉皮层上的神经元是具有多分辨可视、局部响应和方向等特性的，为了模拟这种特性，小波的优势在于其本身具备这些特性。

参考文献

[1.1] Rao C R, Dr. Shalabh, Toutenburg H, et al. Linear Models and Generalizations[M]. 3rd. Germany: Springer, 2007.

[1.2] Boyd, Vandenberghe, Faybusovich. Convex Optimization[J]. IEEE Transactions on Automatic Control, 2006, 51(11): 1859-1859.

[1.3] Lu Z Q J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction[J]. Journal of the Royal Statistical Society: Series A(Statistics in Society), 2010, 27(3): 83-85.

[1.4] Bottou L. Large-scale machine learning with stochastic gradient descent[C]. Proceedings of COMPSTAT'2010. Physica-Verlag HD, 2010: 177-186.

[1.5] Zhang T. Solving large scale linear prediction problems using stochastic gradient descent algorithms [C]. Proceedings of the twenty-first international conference on Machine learning. ACM,

2004：116.

[1.6] Arya S，Mount D M. Algorithms for fast vector quantization[C]. Data Compression Conference. IEEE，2003：381-390.

[1.7] Cox I J，Ghosn J，Yianilos P N，et al. Feature-Based Face Recognition Using Mixture-Distance [C]. Proceedings CVPR '96，1996 IEEE Computer Society Conference on. 1996：209-216.

[1.8] Lecun Y. Generalization and Network Design Strategies[C]. Connectionism in Perspective. 1989.

[1.9] Haykin S. Neural Networks：A Comprehensive Foundation[J]. Neural Networks A Comprehensive Foundation，1994：71-80.

[1.10] Kohonen T. Self-organizing maps[M]. Self-Organizing Maps. Springer Berlin Heidelberg，2001：266-270.

[1.11] Webb A R，Copsey K D. Introduction to Statistical Pattern Recognition[J]. 1972，60：2133-2143.

[1.12] Spiegelhalter D J. Bayesian Analysis in Expert Systems[J]. Statistical Science，1993，8(3)：219-247.

[1.13] Clark P，Boswell R. Rule induction with CN2：Some recent improvements[J]. Lecture Notes in Computer Science，1991，482：151-163.

[1.14] Hopfield J J. Neurons with graded response have collective computational properties like those of two-state neurons[M]. Neurocomputing：foundations of research. American：IT Press，1988：3088-3092.

[1.15] De'Ath G，Fabricius K E. Classification and Regression Trees：A Powerful Yet Simple Technique for Ecological Data Analysis[J]. Ecology，2008，81(11)：3178-3192.

[1.16] 王永庆. 人工智能原理与方法[M]. 西安：西安交通大学出版社，1998.

[1.17] 周志华，陈世福. 神经网络集成[J]. 计算机学报，2002，25(1)：1-8.

[1.18] 焦李成. 神经网络系统理论[M]. 西安：西安电子科技大学出版社，1990.

[1.19] 余凯，贾磊，陈雨强，等. 深度学习的昨天、今天和明天[J]. 计算机研究与发展，2013，50(9)：1799-1804.

[1.20] Jaitly N，Nguyen P，Senior A，et al. Application of Pretrained Deep Neural Networks to Large Vocabulary Conversational Speech Recognition[J]. Proc Interspeech，2012.

[1.21] Lecun Y，Boser B，Denker J，et al. Backpropagation Applied to Handwritten Zip Code Recognition[J]. Neural Computation，1989，1(4)：541-551.

[1.22] Mnih A，Teh Y W. A Fast and Simple Algorithm for Training Neural Probabilistic Language Models[J]. Computer Science，2012：1751-1758.

[1.23] Socher R，Lin C Y，Ng A Y，et al. Parsing Natural Scenes and Natural Language with Recursive Neural Networks[C]. International Conference on Machine Learning，ICML 2011，Bellevue，Washington，USA，June 28-July. DBLP，2011：129-136.

[1.24] Bengio Y. Learning Deep Architectures for AI[J]. Foundations & Trends® in Machine Learning，2009，2(1)：1-127.

[1.25] 张春梅，尹忠科，肖明霞. 基于冗余字典的信号超完备表示与稀疏分解[J]. 科学通报，2006，51(6)：628-633.

[1.26] 焦李成，谭山. 图像的多尺度几何分析：回顾和展望[J]. 电子学报，2003，31(s1)：1975-1981.

[1.27] Cohen A，Daubechies I，Guleryuz O G，et al. On the importance of combining wavelet-based nonlinear approximation with coding strategies[J]. IEEE Transactions on Information Theory，

2002，48(7)：1895-1921.

[1.28] Sturm B. A Wavelet Tour of Signal Processing[M]. A wavelet tour of signal processing. Academic Press，1999：83-85.

[1.29] Mallat S G，Zhang Z. Matching pursuits with time-frequency dictionaries[J]. IEEE Transactions on Signal Processing，1993，41(12)：3397-3415.

[1.30] 汪胜前. 图像的小波稀疏表示及收缩去噪算法[D]. 上海交通大学，2002.

[1.31] Olshausen B A，Field D J. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. [J]. Nature，1996，381(6583)：607.

[1.32] Candes E J. Ridgelets：theory and applications [J]. Icase/LaRC，1998.

[1.33] Do M N，Vetterli M. The contourlet transform：an efficient directional multiresolution image representation[J]. IEEE Transactions on Image Processing，2005，14(12)：2091.

[1.34] Velisavljevic V，Beferull-Lozano B，Vetterli M，et al. Directionlets：anisotropic multidirectional representation with separable filtering[J]. IEEE Transactions on Image Processing，2006，15(7)：1916-1933.

[1.35] Candès E J. Harmonic analysis of neural networks[J]. Applied & Computational Harmonic Analysis，1998，6(2)：197-218.

[1.36] Starck J L，Candes E J，Donoho D L. The curvelet transform for image denoising[J]. IEEE Transactions on Image Processing，2002，11(6)：670-684.

[1.37] Boyd，Vandenberghe，Faybusovich. Convex Optimization[J]. IEEE Transactions on Automatic Control，2006，51(11)：1859-1859.

[1.38] 焦李成.神经网络的应用与实现[M].西安：西安电子科技大学出版社，1992.

[1.39] 焦李成. 神经网络计算[M]. 西安：西安电子科技大学出版社，1992.

[1.40] 焦李成，杨淑媛，刘芳，等. 神经网络七十年：回顾与展望[J]. 计算机学报，2016，39(8)：1697-1716.

[1.41] 焦李成，赵进，杨淑媛，等. 稀疏认知学习、计算与识别的研究进展[J]. 计算机学报，2016，39(4)：835-852.

[1.42] 焦李成，尚荣华，刘芳，杨淑媛，等. 稀疏学习、分类与识别[M].北京：科学出版社，2017.

[1.43] 焦李成. 图像多尺度几何分析理论与应用：后小波分析理论与应用[M]. 西安：西安电子科技大学出版社，2008.

[1.44] 焦李成，谭山. 图像的多尺度几何分析：回顾和展望[J]. 电子学报，2004，31(b12)：1975-1981.

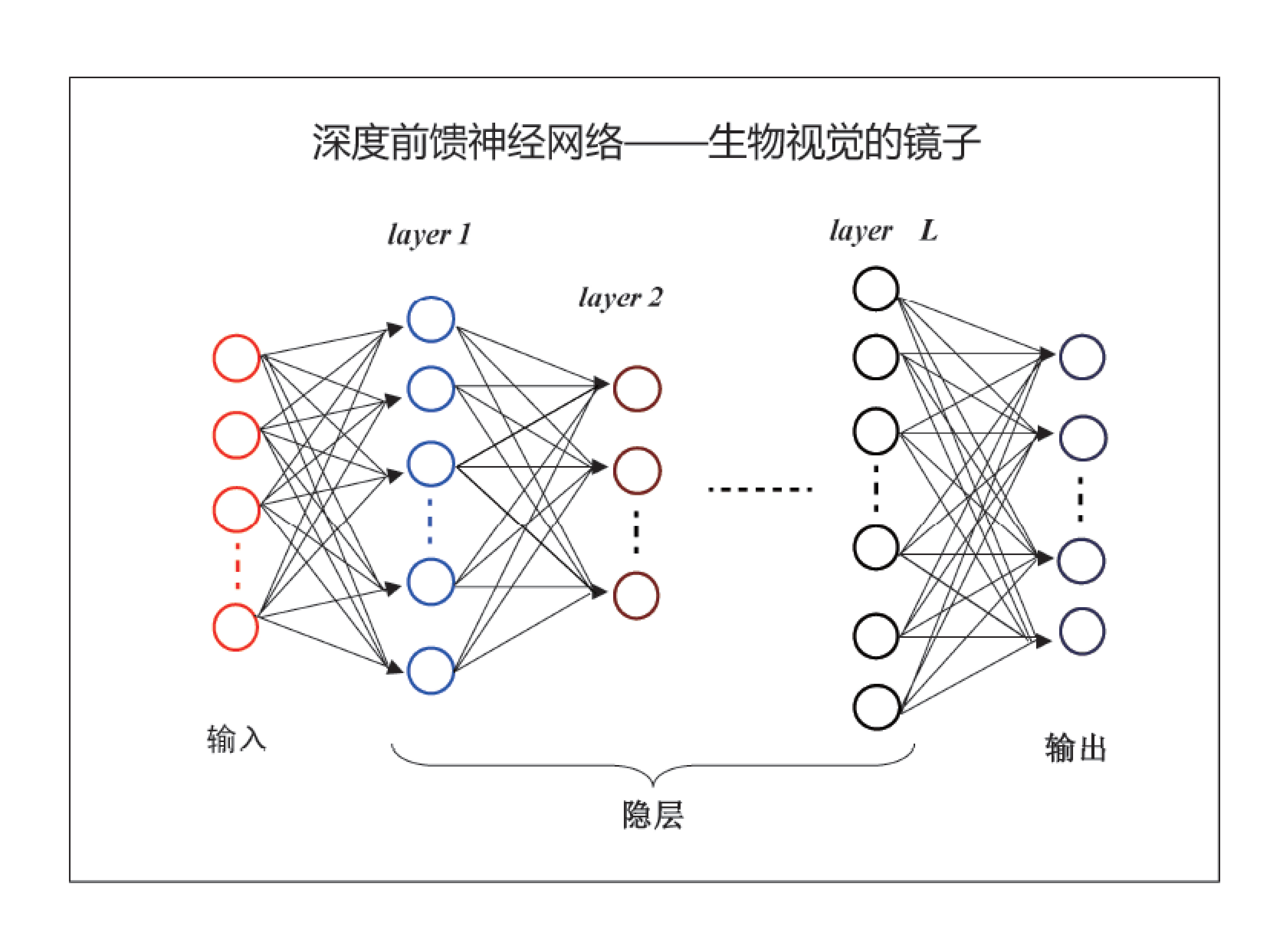
[1.45] 焦李成，周伟达，张莉，等. 智能目标识别与分类[M]. 北京：科学出版社，2010.

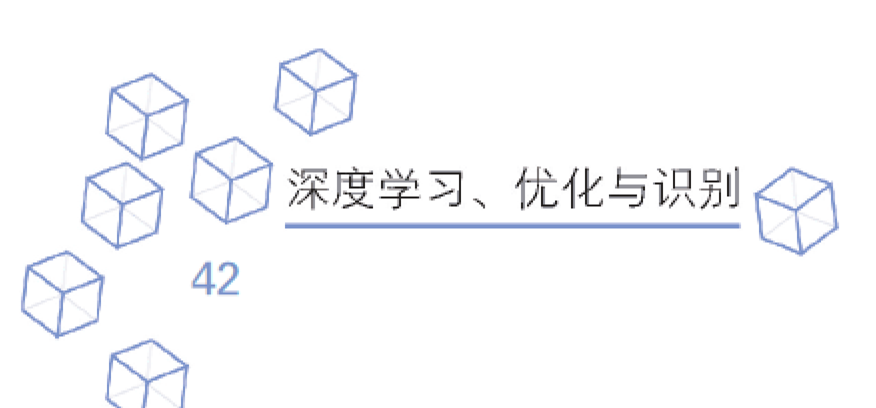
第2章



深度前馈神经网络

CHAPTER 2





2.1　神经元的生物机理

神经元是以生物神经系统的神经细胞为基础的生物模型。在对生物神经系统进行研究时，对神经元的生物机理进行建模，得到基于神经元的计算模型——人工神经网络(或称为神经网络)。在以人工智能为导向的时代，对神经元生物机理(特别是自学习功能)的研究成为神经网络能否在系统辨识、模式识别和智能控制等领域实现技术再次突破的核心。

2.1.1　生物机理

神经元由细胞体、树突和轴突三部分组成，如图 2.1 所示；其中的细胞体是由很多分子形成的综合体，内部含有一个细胞核、核糖体、原生质网状结构等，它是神经元活动的能量供应地，在这里进行新陈代谢等各种生化过程；树突是接收从其他神经元传入信息的入口；轴突是将神经元兴奋信息传出的出口。

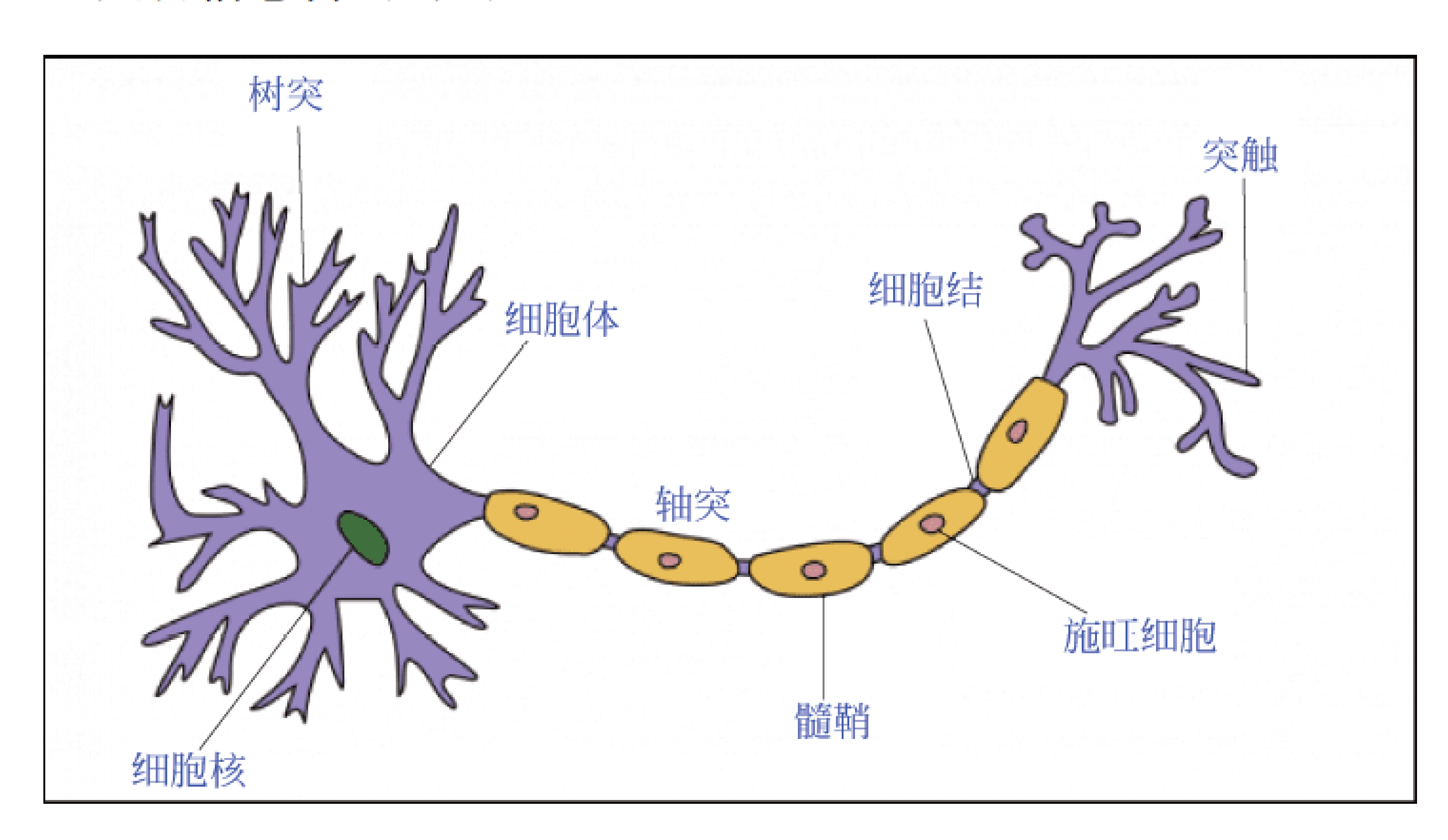


图 2.1　神经元的结构

另外，一个神经元与另一个神经元之间相联系并进行信息传送的结构为突触；注意两个神经元并不直接连通，彼此联系的方式是通过突触这种结构。

目前，根据神经生理学的研究，已发现神经元有 4 种行为，即能处于抑制或兴奋状态、能产生爆发和平台两种情况、能产生抑制后的反冲、具有适应性；另外，突触也有 4 种行为，即能进行信息综合、能产生渐次变化的传送、有电接触和化学接触等多种连接方式、会产生延时激发。关于神经元的信息处理与传递机制具有如下的特性：神经元的抑制和兴奋特性(根据细胞膜内外不同的电位差来表征)、传递的阈值特性(当神经元接收信息时，膜电位逐渐变化，当超出定值时，才会产生脉冲沿轴突传递)、信息综合特性(对不同神经元传递的不同成分的神经递质可在同一个神经元的膜电位内变化并产生作用)、神经元与突触具有数(字)模(拟)转换功能。

但是，大脑中的神经网络非常复杂，尽管解释神经元间线性连接的分子机制已经被描述了很多次，但是对于其神经元分支的工作机制（生物学功能），研究者却并不清楚，例如抑制神经元（如何阻止其他神经元放电）如何执行赢家通吃的策略（这是一种简单的竞争机制，保证了使用频率较高、输入较强的环路连接被保留下来并加以强化，而使频率低、输入较弱的连接被去除，从而使系统资源得到最优化的分配，神经环路的连接更加精确）？大脑网络中这种高度分叉的神经元之间是如何连接的？研究人员表明，包括兴奋神经元以及辅助神经元中的抑制神经元无法证明回路的效率。类似的，任何一种没有察觉收敛神经元和稳定神经元差异的抑制神经元的安排都要比可以察觉到这一区别的安排更低效。那么，假设进化趋于找到解决工程问题的有效解决方案，模型既告诉我们问题的答案就在大脑中，也提出了一个适于经验研究的问题：真实的抑制神经元能否展示出类似收敛神经元和稳定神经元之间的区别（本段参考：学界 MIT 将生物学机制引入神经网络，新模型或揭开抑制神经元功能；Science：科学家揭秘大脑神经元网络形成的复杂机制）。

2.1.2　单隐层前馈神经网络

神经网络是以神经元的数学模型为基础来描述的，其模型由网络拓扑、节点特点和学习规则来表示，其吸引力主要有以下四点：

（1）并行式分布处理；

（2）高度鲁棒性和容错能力；

（3）分布存储及学习能力；

（4）能充分逼近复杂的非线性关系。

下面根据神经元的特性和生物学功能可以知道，神经元是一个多输入单输出的信息处理单元，而且，它对信息的处理是非线性的，将其抽象为一个简单的数学模型如图 2.2 所示。

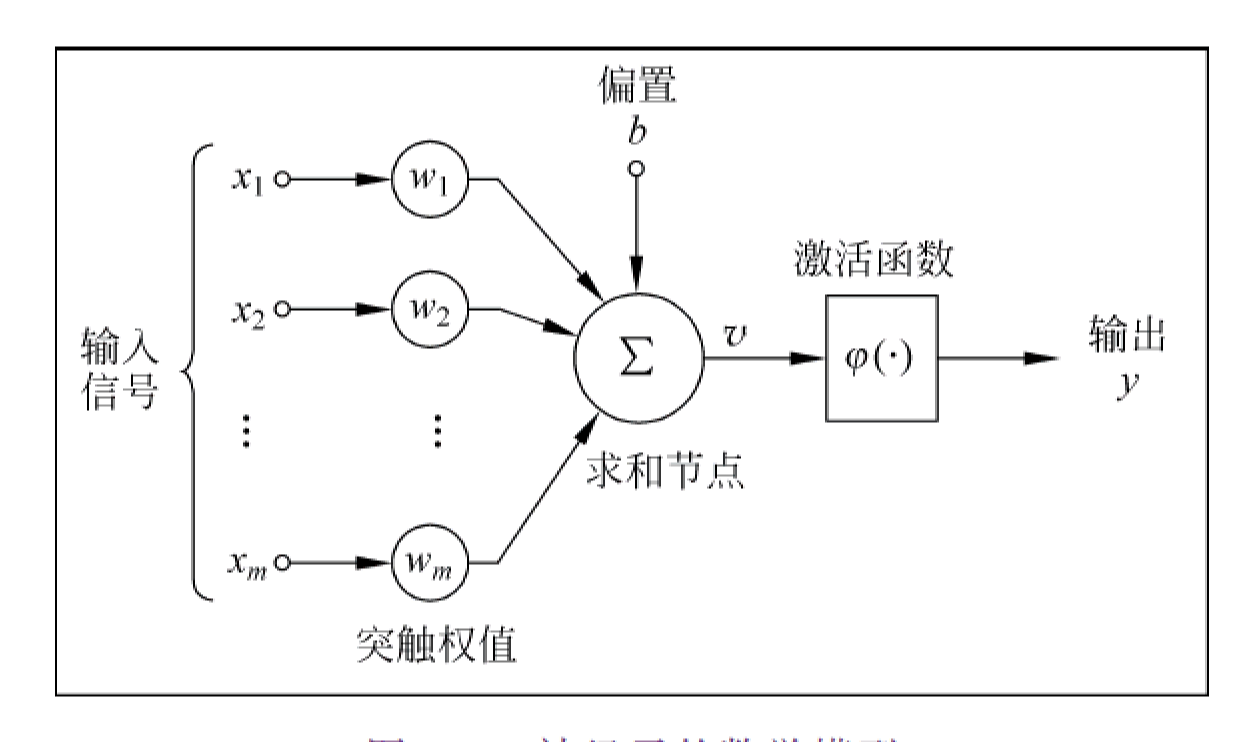


图 2.2　神经元的数学模型

具体的数学公式如下：

$$
\begin{cases}
v = \sum_{i=1}^{m} x_i w_i + b \\
y = \varphi(v)
\end{cases}
\tag{2.1}
$$

典型的激活函数有 sigmoid 函数、tanh 函数、径向基函数、小波函数、修正线性单元(ReLU)、softplus 函数等,对应的公式为:

$$\begin{cases} \text{sigmoid}(x)=\dfrac{1}{1+e^{-x}} \\ \tanh(x)=\dfrac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \\ \text{ReLU}(x)=\max(0,x) \\ \text{softplus}(x)=\log(1+e^{x}) \end{cases} \tag{2.2}$$

值得注意的是,神经科学家发现神经元具有单侧抑制、宽兴奋边界、稀疏激活等特性。与其他激活函数相比,修正线性单元具有生物上的可解释性(即描述神经元的这些特性)。另外,softplus 函数的导数为 logistics 函数,它的名称源于它是修正线性单元的平滑形式,虽然它也具有单侧抑制、宽兴奋边界特性,但是没有稀疏激活特性。

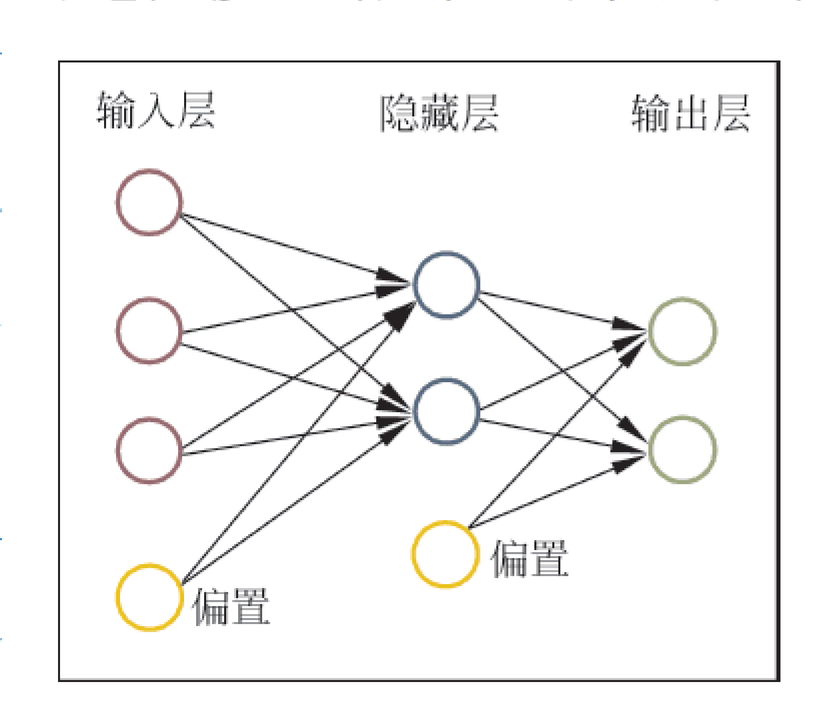


图 2.3　单隐层前馈神经网络

基于神经元的数学模型,根据网络连接的拓扑结构,神经网络模型可以分为前向网络(有向无环)和反馈网络(无向完备图,也称循环网络),对于反馈网络,网络模型的稳定性与联想记忆有着密切的关系,Hopfield 网络、玻尔兹曼机网络都属于这种类型。而对于前向网络,源于简单非线性函数的多次复合,网络结构简单,易于实现。下面主要介绍前向网络(单隐层前馈神经网络),其网络结构如图 2.3 所示。对应的数学公式为:

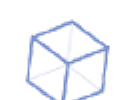
$$\begin{cases} \boldsymbol{h}^{(1)}=\varphi^{(1)}\left(\sum_{i=1}^{m}\boldsymbol{x}_i\cdot\boldsymbol{w}_i^{(1)}+\boldsymbol{b}^{(1)}\right) \\ \boldsymbol{y}=\varphi^{(2)}\left(\sum_{j=1}^{n}\boldsymbol{h}_j^{(1)}\cdot\boldsymbol{w}_j^{(2)}+\boldsymbol{b}^{(2)}\right) \end{cases} \tag{2.3}$$

其中输入 $\boldsymbol{x}\in\mathbb{R}^m$,隐层输出 $\boldsymbol{h}\in\mathbb{R}^n$,输出 $\boldsymbol{y}\in\mathbb{R}^K$,$\boldsymbol{w}^{(1)}\in\mathbb{R}^{m\times n}$ 与 $\boldsymbol{b}^{(1)}\in\mathbb{R}^n$ 分别为输入到隐藏层的权值连接矩阵和偏置,$\boldsymbol{w}^{(2)}\in\mathbb{R}^{n\times K}$ 与 $\boldsymbol{b}^{(2)}\in\mathbb{R}^K$ 分别为隐藏层到输出层的权值连接矩阵和偏置,$\varphi^{(1)}$ 和 $\varphi^{(2)}$ 为相应的激活函数。

实际应用中,假设训练数据集为:

$$\begin{cases} \{\boldsymbol{x}^{(n)},\boldsymbol{y}^{(n)}\}_{n=1}^{N} \\ \boldsymbol{x}^{(n)}\in\mathbb{R}^m \\ \boldsymbol{y}^{(n)}\in\mathbb{R}^K \end{cases} \tag{2.4}$$

输入与输出之间的模型为公式(2.3),即



$$\boldsymbol{y} = T(\boldsymbol{x},\theta) = \varphi^{(2)}\left(\sum_{j=1}^{n}\varphi^{(1)}\left(\sum_{i=1}^{m}\boldsymbol{x}_i \cdot \boldsymbol{w}_i^{(1)} + \boldsymbol{b}^{(1)}\right)\cdot \boldsymbol{w}_j^{(2)} + \boldsymbol{b}^{(2)}\right) \tag{2.5}$$

其中的参数 $\theta=(\boldsymbol{w}^{(1)},\boldsymbol{b}^{(1)};\boldsymbol{w}^{(2)},\boldsymbol{b}^{(2)})$，进一步优化目标(损失项和正则项构成)为：

$$\min_{\theta} L(\theta) = \frac{1}{N}\sum_{n=1}^{N}\|\boldsymbol{y}^{(n)} - T(\boldsymbol{x}^{(n)};\theta)\|_F^2 + \lambda\sum_{l=1}^{2}\|\boldsymbol{w}^{(l)}\|_F^2 \tag{2.6}$$

通过梯度下降的方法，求解参数 θ，即

$$\begin{cases}\theta^k = \theta^{k-1} - \alpha \cdot ?\theta\,|_{\theta=\theta^{k-1}} \\ ?\theta\,|_{\theta=\theta^{k-1}} = \left.\dfrac{\partial L(\theta)}{\partial\theta}\right|_{\theta=\theta^{k-1}}\end{cases} \tag{2.7}$$

随着迭代次数 k 的增加，参数将收敛(间接可通过目标函数 $L(\theta_k)$来可视化进行观察)，即

$$\lim_{k\to\infty}\theta^k = \theta^* \tag{2.8}$$

收敛的原因是因为上述目标函数为凸的。注意，对于优化目标函数(2.6)，本可以直接利用闭形式解直接求出，但是数据量大的时候，存储及读取将会非常耗时，所以通常利用随机梯度下降(即批量化的处理)来求解。对于神经网络拓扑结构的确定，Hornik 等人已证明：若输出层采用线性激活函数，隐层采用 Sigmoid 函数，则单隐层神经网络能够以任意精度逼近任何有理函数。

2.2　多隐层前馈神经网络

沿用单隐层前馈神经网络的分析，当隐层个数超过 2 层(包括两层)时，称为多隐层前馈神经网络，或深度前馈神经网络，其网络结构如图 2.4 所示。

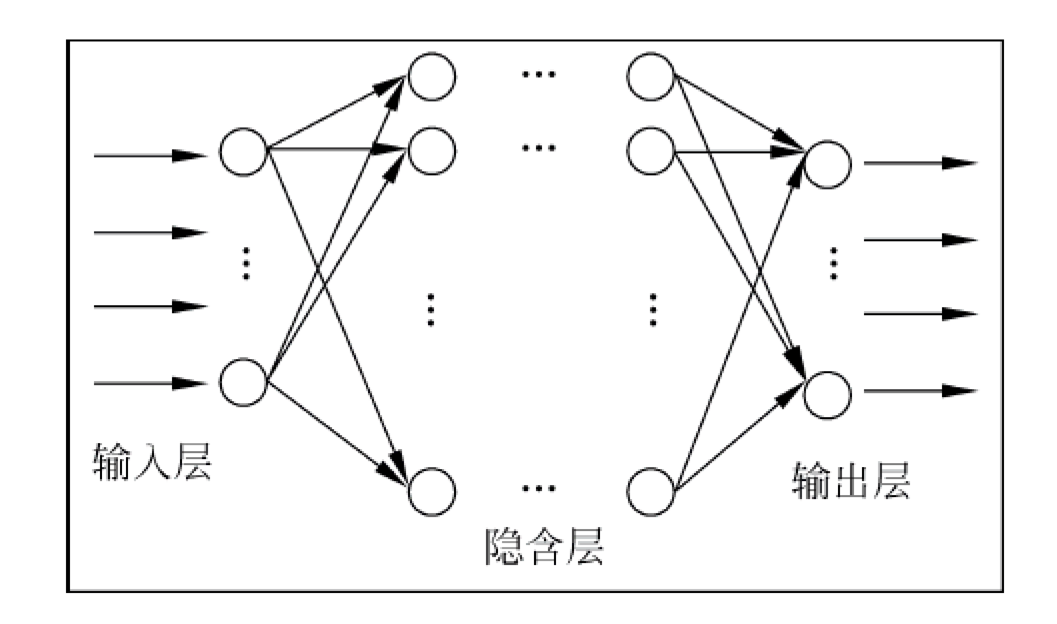


图 2.4　多隐层前馈神经网络

这里需要指出的是深度前馈神经网络的拓扑结构是：多隐层、全连接且有向无环。基于图 2.4，利用下面的记号，给出网络输入与输出之间的模型：

输入 $\boldsymbol{x}\in\mathbb{R}^m$，输出 $\boldsymbol{y}\in\mathbb{R}^s$，隐层的输出记为：

$$
\begin{cases}
\boldsymbol{h}^{(l)} = \varphi^{(l)}\left(\sum_{i=1}^{n_{l-1}} \boldsymbol{h}_i^{(l-1)} \boldsymbol{w}_i^{(l)} + \boldsymbol{b}^{(l)}\right) \\
l = 1,2,\cdots,L \\
\boldsymbol{h}^{(0)} = \boldsymbol{x} \\
\boldsymbol{h}^{(L)} = \boldsymbol{y}
\end{cases} \tag{2.9}
$$

需要注意的是，除去输入层 $\boldsymbol{h}^{(0)}$ 与输出层 $\boldsymbol{h}^{(L)}$，隐层的个数共计 $L-1$ 层，对应的超参数（层数、隐单元个数、激活函数）为：

$$
\begin{cases}
L+1 \quad \rightarrow \text{层数（包含输入与输出）} \\
[n_0, n_1, n_2, \cdots, n_{L-1}, n_L] \quad \rightarrow \text{每一层上的维数} \\
[\varphi^{(1)}, \varphi^{(2)}, \cdots, \varphi^{(L-1)}, \varphi^{(L)}] \quad \rightarrow \text{激活函数}
\end{cases} \tag{2.10}
$$

注意 $n_0=m$ 和 $n_L=s$，并且待学习的参数记为：

$$
\begin{cases}
\theta = (\theta_1, \theta_2, \cdots, \theta_L) \\
\theta_l = (\boldsymbol{w}^{(l)} \in \mathbb{R}^{n_{l-1}\times n_l}, \boldsymbol{b}^{(l)} \in \mathbb{R}^{n_l}) \\
l = 1,2,\cdots,L
\end{cases} \tag{2.11}
$$

那么输入与输出的关系为：

$$
\begin{aligned}
\boldsymbol{y} = \boldsymbol{h}^{(L)} &= \varphi^{(L)}\left(\sum_{i_L=1}^{n_L} \boldsymbol{h}_{i_L}^{(L-1)} \cdot \boldsymbol{w}_{i_L}^{(L)} + \boldsymbol{b}^{(L)}\right) \quad \rightarrow \text{记为 } \varphi^{(L)}(h^{(L-1)}, \theta_L) \\
&= \varphi^{(L)}\left(\sum_{i_L=1}^{n_L} \varphi^{(L-1)}\left(\sum_{i_{L-1}=1}^{n_{L-1}} \boldsymbol{h}_{i_{L-1}}^{(L-2)} \cdot \boldsymbol{w}_{i_{L-1}}^{(L-1)} + \boldsymbol{b}^{(L-1)}\right)\boldsymbol{w}_{i_L}^{(L)} + \boldsymbol{b}^{(L)}\right) \\
&\qquad \rightarrow \text{记为 } \varphi^{(L)}(\varphi^{(L-1)}(h^{(L-2)}, \theta_{L-1}), \theta_L) \\
&= \cdots = \varphi^{(L)}(\varphi^{(L-1)}(\cdots\varphi^{(1)}(\boldsymbol{x}, \theta_1)\cdots, \theta_{L-1}), \theta_L) \quad \rightarrow \text{记为 } f(x, \theta)
\end{aligned} \tag{2.12}
$$

实际应用中，对于训练数据集：

$$
\begin{cases}
\{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N} \\
\boldsymbol{x}^{(n)} \in \mathbb{R}^m \\
\boldsymbol{y}^{(n)} \in \mathbb{R}^s
\end{cases}
$$

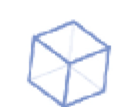
所得到的优化目标函数（损失项和正则项构成）：

$$
\min_{\theta} J(\theta) = L(\theta) + \lambda R(\theta) \tag{2.13}
$$

其中 $\hat{\boldsymbol{y}}_n = f(\boldsymbol{x}_n, \theta)$ 以及：

$$
\begin{cases}
l(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n) = \| \boldsymbol{y}_n - \hat{\boldsymbol{y}}_n \|_F^2 \\
L(\theta) = \dfrac{1}{N}\sum_{n=1}^{N} l(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n) \\
R(\theta) = \sum_{l=1}^{L} \| \theta_l \|_F^2 = \sum_{l=1}^{L} \| \boldsymbol{w}^{(L)} \|_F^2
\end{cases} \tag{2.14}
$$

注意损失函数 $l(\cdot)$ 有很多形式：能量损失，交叉熵损失等；正则项 $R(\cdot)$ 除了利用富比尼



斯范数(防止过拟合)外,还有稀疏正则(模拟生物响应特性),此处不再赘述,后续章节会有所介绍。

2.3 反向传播算法

针对优化目标函数公式(2.13),如何求解? 首先,要确定目标函数的凸性与非凸性,如果可行域(即参数的选取范围)是凸集的话,定义在该凸集上的凸函数为凸优化,即求得的解不依赖于初值的选取且为全局最优解;通常,深度前馈神经网络的优化目标函数为非凸的,所以参数的求解依赖于初始参数的设置(即可行域内存在大量的鞍点与局部极值点),如果设置合理,可以避免过早陷入局部最优;其次,为了说明反向传播算法(基于梯度下降方法),具体描述如下,即通过如下的方法来更新参数:

$$\begin{cases}\theta^{(k)}=\theta^{(k-1)}-\alpha\cdot ?\theta\mid_{\theta=\theta^{(k)}}\\ ?\theta\mid_{\theta=\theta^{(k)}}=\dfrac{\partial L(\theta)}{\partial\theta}+\lambda\dfrac{\partial R(\theta)}{\partial\theta}\end{cases}\tag{2.15}$$

其中 α 为学习速率,具体的关于每一层上的参数更新为:

$$\begin{cases}\theta_l^{(k)}=\theta_l^{(k-1)}-\alpha\cdot ?\theta_l\mid_{\theta_l=\theta_l^{(k-1)}}\\ ?\theta_l\mid_{\theta_l=\theta_l^{(k-1)}}=\dfrac{\partial L(\theta)}{\partial\theta_l}\Big|_{\theta_l=\theta_l^{(k-1)}}+\lambda\dfrac{\partial R(\theta)}{\partial\theta_l}\Big|_{\theta_l=\theta_l^{(k-1)}}\end{cases}\tag{2.16}$$

这里,$\theta_l^{(k)}$ 为第 l 层第 k 次迭代的更新,对于核心梯度下降量的求解,引入误差传播项,根据链式法则,将其展开为:

$$\frac{\partial L(\theta)}{\partial\theta_l}=\frac{\partial\boldsymbol{h}^{(l)}}{\partial\theta_l}\cdot\frac{\partial\boldsymbol{h}^{(l+1)}}{\partial\boldsymbol{h}^{(l)}}\cdot\cdots\cdot\frac{\partial\boldsymbol{h}^{(L)}}{\partial\boldsymbol{h}^{(L-1)}}\cdot\frac{\partial L(\theta)}{\partial\boldsymbol{h}^{(L)}}\tag{2.17}$$

其中误差传播项记为:

$$\boldsymbol{\delta}^{(l)}=\frac{\partial L(\theta)}{\partial\boldsymbol{h}^{(l)}}\tag{2.18}$$

进一步利用 $\theta_l=(\boldsymbol{w}^{(l)},\boldsymbol{b}^{(l)})$,则隐层输出关于对应的参数求导为:

$$\begin{cases}\dfrac{\partial\boldsymbol{h}^{(l)}}{\partial\boldsymbol{w}^{(l)}}=\dfrac{\partial\varphi^{(l)}((\boldsymbol{h}^{l-1})^{\mathrm{T}}\cdot\boldsymbol{w}^{(l)}+\boldsymbol{b}^{(l)})}{\partial\boldsymbol{w}^{(l)}}=\boldsymbol{h}^{l-1}\circ(\varphi^{(l)})'\\ \dfrac{\partial\boldsymbol{h}^{(l)}}{\partial\boldsymbol{b}^{(l)}}=\dfrac{\partial\varphi^{(l)}((\boldsymbol{h}^{l-1})^{\mathrm{T}}\cdot\boldsymbol{w}^{(l)}+\boldsymbol{b}^{(l)})}{\partial\boldsymbol{b}^{(l)}}=\boldsymbol{1}\circ(\varphi^{(l)})'\end{cases}\tag{2.19}$$

其中$\circ$为 Hadamard 积。从公式(2.17)可以看出,注意公式(2.17)是损失项关于参数的求导,而正则项关于参数的导数为:

$$\frac{\partial R(\theta)}{\partial\theta_l}=\frac{\partial}{\partial\theta_l}\left(\sum_{l=1}^{L}\|\theta_l\|_F^2\right)=\frac{\partial\|\theta_l\|_F^2}{\partial\theta_l}\tag{2.20}$$

通常,正则项中的约束仅仅针对权值矩阵,而偏置不加正则约束,所以有:

$$\begin{cases}\dfrac{\partial R(\theta)}{\partial \boldsymbol{w}^{(l)}}=\dfrac{\partial \|\boldsymbol{w}^{(l)}\|_F^2}{\partial \boldsymbol{w}^{(l)}}=2\boldsymbol{w}^{(l)} \\ \dfrac{\partial R(\theta)}{\partial \boldsymbol{b}^{(l)}}=\dfrac{\partial \|\boldsymbol{w}^{(l)}\|_F^2}{\partial \boldsymbol{b}^{(l)}}=\mathbf{0}\end{cases} \tag{2.21}$$

注意：反向传播算法的核心含义是：即优化目标函数 $J(\theta)$ 中关于第 l 个隐层参数 θ_l 的梯度下降量，分别由损失项 $L(\theta)$ 和正则项 $R(\theta)$ 关于第 l 个隐层参数 θ_l 的梯度（一阶导数）决定，其中通过引入误差传播项（式(2.18)）来实现误差的反向传播。所以前馈神经网络的训练分为两步：一是根据当前的参数值，计算前向传播过程中每一层上的输出值；二是根据实际输出与期望输出之间的差来反向传播计算每一层上的误差传播项，结合每一层输出关于该层参数的偏导数，实现每一层参数的更新；重复这两步，直至该过程收敛（图 2.5 所示）。注意的是，当网络层数很深的时候，误差关于每一层上参数的梯度下降量会随着输出到输入端的传播过程逐渐衰减（即越靠近输出端，下降量越大，越靠近输入端，下降量越小，甚至下降量几乎为零），使得整个网络很难通过训练获取较好的层级参数，从而避免可行域上的鞍点与局部极值点，往往陷入局部最优，这就是梯度弥散问题。

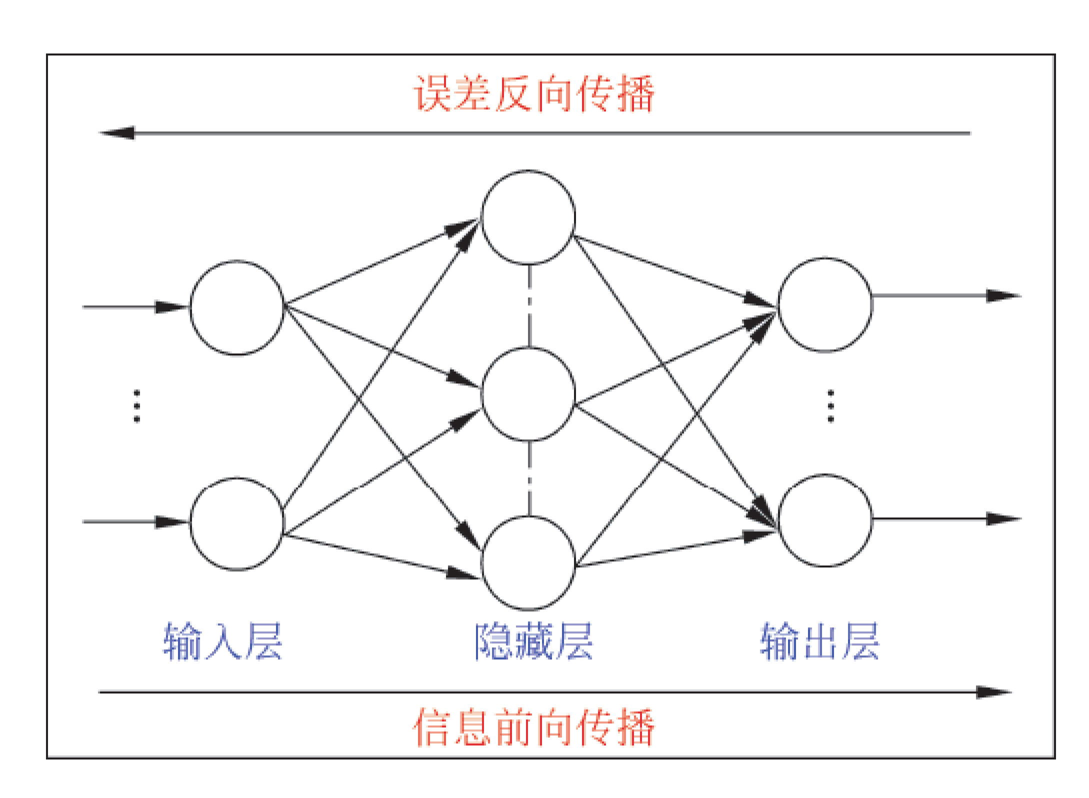
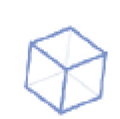


图 2.5　反向传播算法的图示

2.4　深度前馈神经网络的学习范式

深度前馈神经网络仍沿用机器学习的范式，即数据、模型、优化和求解 4 个部分。机器学习强调基于数据先验的特征学习（包括特征提取与筛选，得到可分性判别特征）与分类器的设计，并且模型的表达能力受限于（统计或变换，本质上为浅层）特征学习，优势在于优化目标函数可利用凸优化相关算法或软件快速地求解，其核心理念在于追求精度、速度。图 2.6 为特征学习：可分性判别特征的学习图示。

相较于机器学习，深度前馈神经网络减小了对数据先验的依赖性，模型对数据的表征能力（挖掘数据深层的语义信息或统计特性）随着层级的加深（线性与非线性逐层复合）而呈现愈来愈深刻、本质的刻画；同时模型的缺点有：



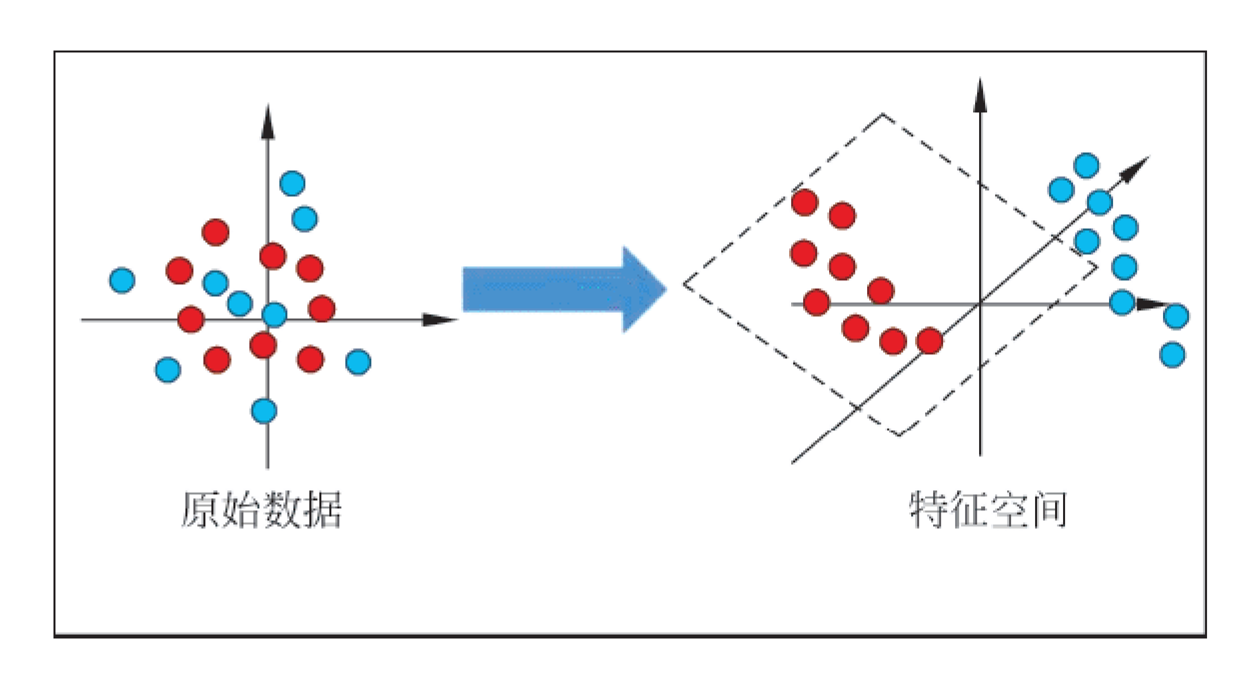


图 2.6　特征学习：可分性判别特征的学习

（1）训练阶段，有类标数据较少，网络模型参数较多，训练不充分，易出现过拟合现象。

（2）优化目标函数为非凸优化问题，依赖于初值的选取。选择较好时，可以避免过早地陷入局部最优，求得的解逼近最优解；若选择不好时，网络易出现欠拟合，如图 2.7 所示。

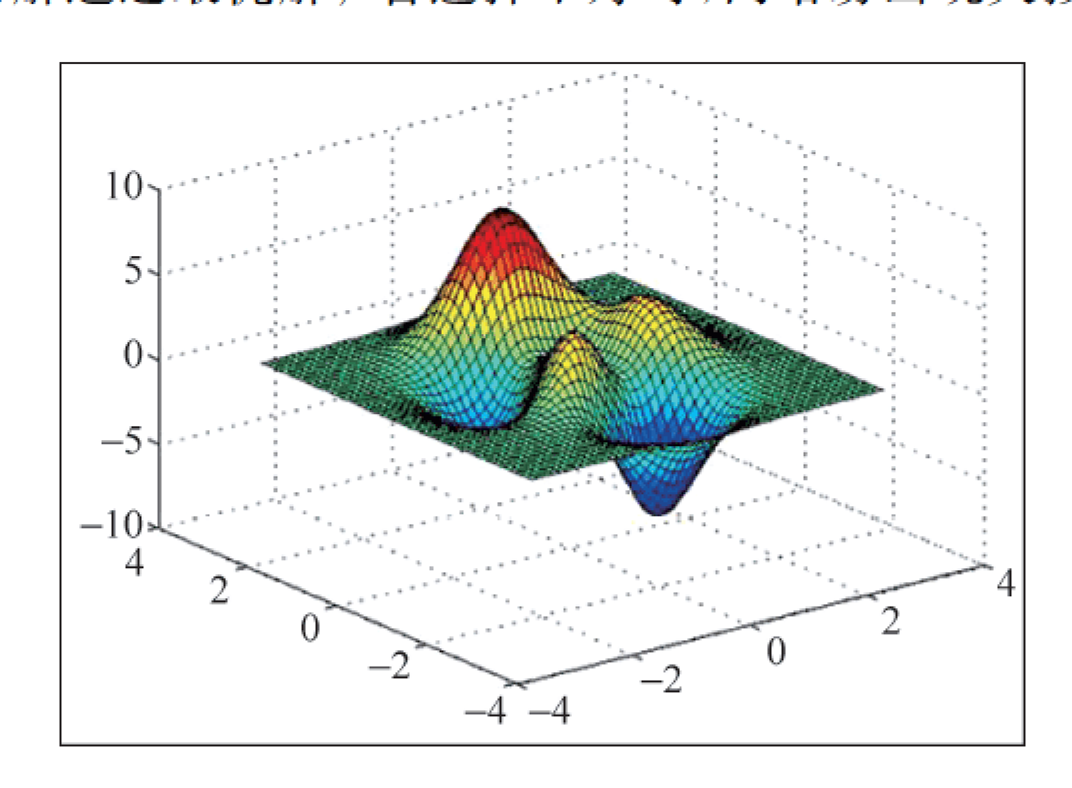


图 2.7　可视化非凸优化问题——局部极小值

（3）利用反向传播算法优化求解时，易出现梯度弥散的现象，导致网络模型训练不充分（参数更新时效性和有效性差）。

众所周知，数据的差异性对深度前馈神经网络的影响是至关重要的，譬如分类任务，类内的聚集特性越强，说明相似度越高，即共性特征占主要，个性化特征为辅；类间的疏散特性越大，说明类与类之间的差异性越明显，即个性化特性为主，共性特征为辅；对于利用深度前馈神经网络进行特征学习而言，层级参数的组合多样性、容许性强，使得权值参数带有判别特性，即类内强调共性，类间注重个性。参数组合下满意度最高的模型状态也间接说明二者（共性与个性）是矛盾统一的。本质上，深度前馈神经网络将数据的表示分级，高级的表示建立在低级的表示上，即将一个复杂的问题分成一系列嵌套的、简单的表示学习问题；例如第一个隐层从图像的像素和邻近像素的像素值中识别边缘；第二个隐层将边缘整合起来识别轮廓和角点；第三个隐层提取特定的轮廓和角点作为抽象的高层语义特征；最后通过一个线性分类器识别图像中的目标。

从物理角度，深度前馈神经网络所有数学运算（包括线性和非线性）的意义在于以下5种形式：升维或降维、放大或缩小、旋转、平移、扭曲或弯曲（非线性操作完成，不同的激活函数对输入的扭曲程度不同）；即每层神经网络的物理释义为：通过现有的不同物质的组合形成新物质，例如碳氧原子通过不同组合形成若干分子，从分子层面继续迭代这种组合思想，可以形成DNA、细胞、组织、器官，最终可以形成一个完整的人；同样的，继续迭代还会有家庭、公司、国家等，这种现象在身边随处可见。

从实验角度观察，对于深度前馈神经网络的模型架构具有以下的特点。

（1）线性可分视角：深度前馈神经网络的学习就是学习如何利用线性变换和非线性变换（激活函数），将输入空间投向线性可分/稀疏的空间去分类/回归。

（2）增加节点数：增加维度，即增加线性转换能力。

（3）增加层数：增加激活函数的次数，即增加非线性转换次数。

为了满足以上的特点，使得学到的特征具有可分特性，见图2.8。

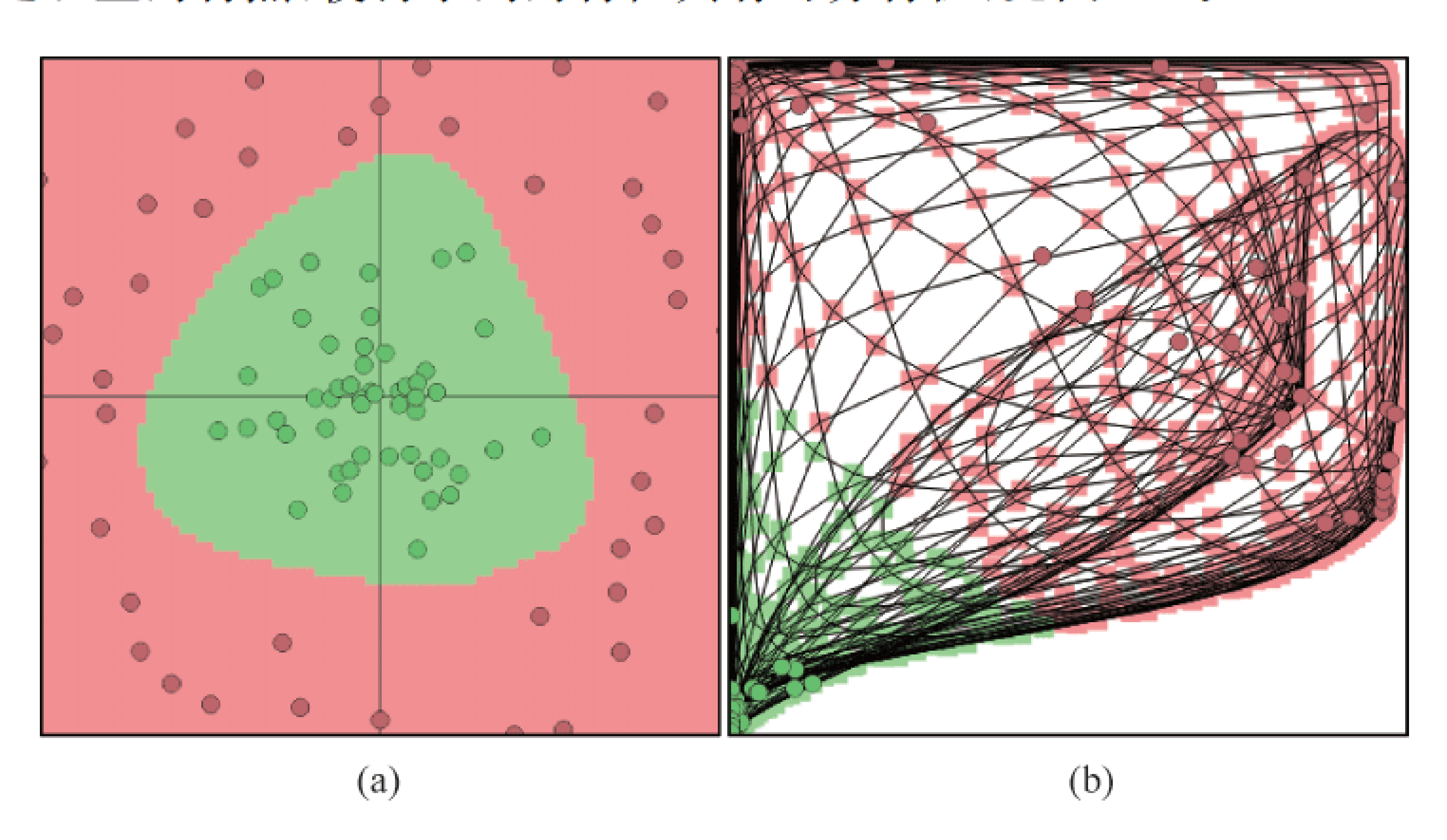


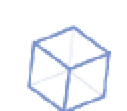
图2.8　两个隐层前馈神经网络的特征学习可视化

其中的两个隐层前馈神经网络模型结构为：

```
layer_defs = [];
layer_defs.push({type: 'input',out_sx:1,out_sy:1,out_depth:2});
layer_defs.push({type:'fc',num_neurons:6,activation:| 'tanh'});
layer_defs.push({type:'fc',num_neurons:2,activation:'tanh'});
layer_defs.push({type:'softmax',num_classes:2});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnet js.SGDTrainer(net,{learning_rate:0.01,momentum:0.1,batch_size:10,12_
decay:0.001});
```



激活函数为 tanh；采用随机梯度下降的方式求解，分类器为 softmax；从图 2.8 可知，图 2.8(a)为原始数据所在空间的可视化；通过线性和非线性操作的两次变换，得到特征空间下的可分性图 2.8(b)。

关于数据随着网络深度的变化，而呈现可分性表达的能力，可参考备注网站：http://cs.stanford.edu/people/karpathy/convnetjs//demo/classify2d.html.

参考文献

[2.1] Lee T S, Mumford D, Romero R, et al. The role of the primary visual cortex in higher level vision[J]. Vision Research, 1998, 38(15-16): 2429.

[2.2] Hebb D O. The organization of behavior[J]. Journal of Applied Behavior Analysis, 1949, 25(3): 575-577.

[2.3] Eccles J C, Fatt P, Koketsu K. Cholinergic and inhibitory synapses in a pathway from motor-axon collaterals to motoneurones[J]. Journal of Physiology, 1954, 126(3): 524.

[2.4] Stoerig P. Blindsight, conscious vision, and the role of primary visual cortex[J]. Progress in Brain Research, 2006, 155: 217-234.

[2.5] Malik J, Perona P. Preattentive texture discrimination with early vision mechanisms[J]. Journal of the Optical Society of America A Optics & Image Science, 1990, 7(5): 923-32.

[2.6] Krüger N, Janssen P, Kalkan S, et al. Deep hierarchies in the primate visual cortex: what can we learn for computer vision? [J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2013, 35(8): 1847-1871.

[2.7] Fukushima K. Artificial vision by multi-layered neural networks: neocognitron and its advances[J]. Neural Networks the Official Journal of the International Neural Network Society, 2013, 37(1): 103.

[2.8] Hubel D H, Wiesel T N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex[J]. Journal of Physiology, 1962, 160(1): 106.

[2.9] Riesenhuber M, Poggio T. Hierarchical models of object recognition in cortex[J]. Nature Neuroscience, 1999, 2(11): 1019-1025.

[2.10] Bruce C, Desimone R, Gross C G. Visual properties of neurons in a polysensory area in superior temporal sulcus of the macaque[J]. Journal of Neurophysiology, 1981, 46(2): 369-384.

[2.11] Fujita I, Tanaka K, Ito M, et al. Columns for visual features of objects in monkey inferotemporal cortex[J]. Nature, 1992, 360(6402): 343-346.

[2.12] Hinton G E. How neural networks learn from experience[J]. Scientific American, 1992, 267(3): 144.

[2.13] Rosenblatt F. The perceptron: A probabilistic model for information storage and organization in the brain[J]. Psychological Review, 1958, 65(6): 386-408.

[2.14] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators[J]. Neural Networks, 1989, 2(5): 359-366.

[2.15] Hornik K. Approximation capabilities of multilayer feedforward networks[J]. Neural Networks,

1991，4(2)：251-257.

[2.16] Hornik K，Stinchcombe M，White H. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks[J]. Neural Networks，1990，3(5)：551-560.

[2.17] Gardner M W，Dorling S R. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences[J]. Atmospheric Environment，1998，32(14-15)：2627-2636.

[2.18] Pinkus A. Approximation theory of the MLP model in neural networks[J]. Acta Numerica，1999，8：143-195.

[2.19] Ze H，Senior A，Schuster M. Statistical parametric speech synthesis using deep neural networks[C]. IEEE International Conference on Acoustics，Speech and Signal Processing. IEEE，2013：7962-7966.

[2.20] Zhu Q，Stolcke A，Chen B Y，et al. Using MLP features in SRI's conversational speech recognition system. [C]. INTERSPEECH 2005—Eurospeech，European Conference on Speech Communication and Technology，Lisbon，Portugal，September. DBLP，2005：2141-2144.

[2.21] Cardinaux F，Sanderson C，Marcel S. Comparison of MLP and GMM Classifiers for Face Verification on XM2VTS[J]. 2003，2688：911-920.

[2.22] Yang S E，Huang L. Financial Crisis Warning Model based on BP Neural Network[J]. Systems Engineering-theory & Practice，2005，25(1)：12-19.

[2.23] Hagan M T，Menhaj M B. Training feedforward networks with the Marquardt algorithm[J]. IEEE Transactions on Neural Networks，1994，5(6)：989-993.

[2.24] Montana D J，Davis L. Training feedforward neural networks using genetic algorithms[C]. International Joint Conference on Artificial Intelligence. Morgan Kaufmann Publishers Inc. 1989：762-767.

[2.25] Murray A F，Edwards P J. Synaptic Weight Noise During MLP Learning Enhances Fault-Tolerance，Generalization and Learning Trajectory[J]. IEEE Trans on Neural Networks，1996：491-498.

[2.26] Rumelhart D E，Hinton G E，Williams R J. Learning internal representation by back-propagation of errors[J]. Nature，1986，323(323)：533-536.

[2.27] Hush D R，Salas J M. Improving the learning rate of back-propagation with the gradient reuse algorithm[C]. IEEE International Conference on Neural Networks. IEEE，1988：441-447 vol. 1.

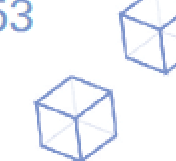
[2.28] Ochiai K，Toda N，Usui S. Kick-out learning algorithm to reduce the oscillation of weights[J]. Neural Networks，1994，7(5)：797-807.

[2.29] Bengio Y，Lamblin P，Popovici D，et al. Greedy layer-wise training of deep networks[C]. International Conference on Neural Information Processing Systems. MIT Press，2006：153-160.

[2.30] 刘曙光，郑崇勋，刘明远. 前馈神经网络中的反向传播算法及其改进：进展与展望[J]. 计算机科学，1996，23(1)：76-79.

[2.31] Yu X H，Chen G A. Efficient Backpropagation Learning Using Optimal Learning Rate and Momentum[J]. Journal of Southeast University，1997，10(3)：517-527.

[2.32] Yu X H，Chen G A，Cheng S X. Acceleration of backpropagation learning using optimised learning rate and momentum[J]. Electronics Letters，1993，29(14)：1288-1290.



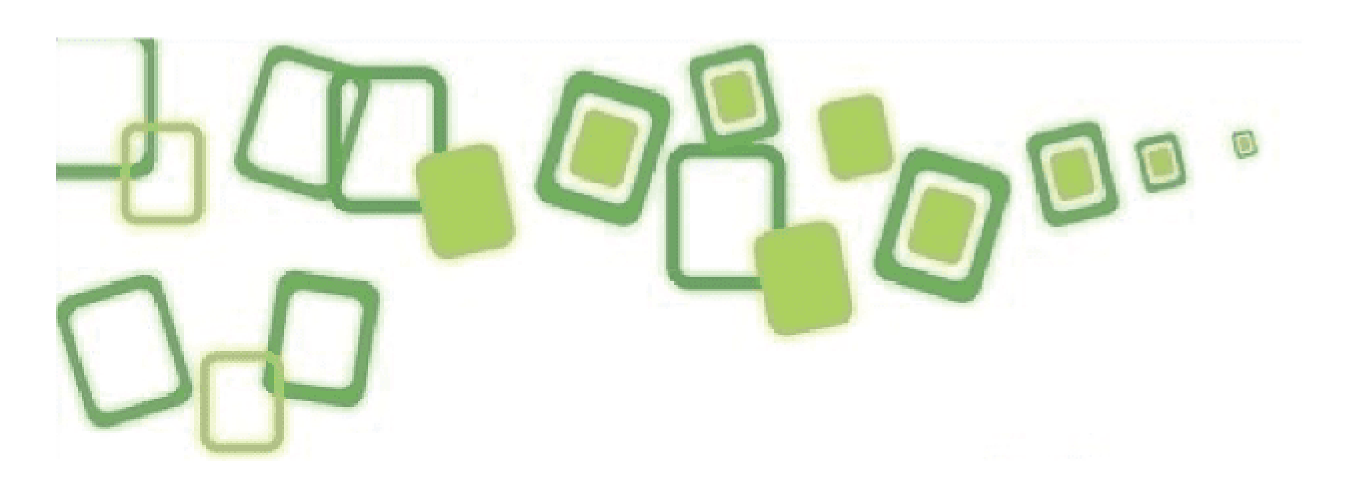
[2.33] Yu X H，Chen G A，Cheng S X. Dynamic learning rate optimization of the backpropagation algorithm[J]. IEEE Transactions on Neural Networks，1995，6(3)：669-677.

[2.34] Magoulas G D，Vrahatis M N，Androulakis G S. Improving the convergence of the backpropagation algorithm using learning rate adaptation methods[J]. Neural Computation，2006，11(7)：1769.

[2.35] Murray A F，Edwards P J. Synaptic Weight Noise During MLP Learning Enhances Fault-Tolerance，Generalization and Learning Trajectory[J]. IEEE Trans on Neural Networks，1992：491-498.

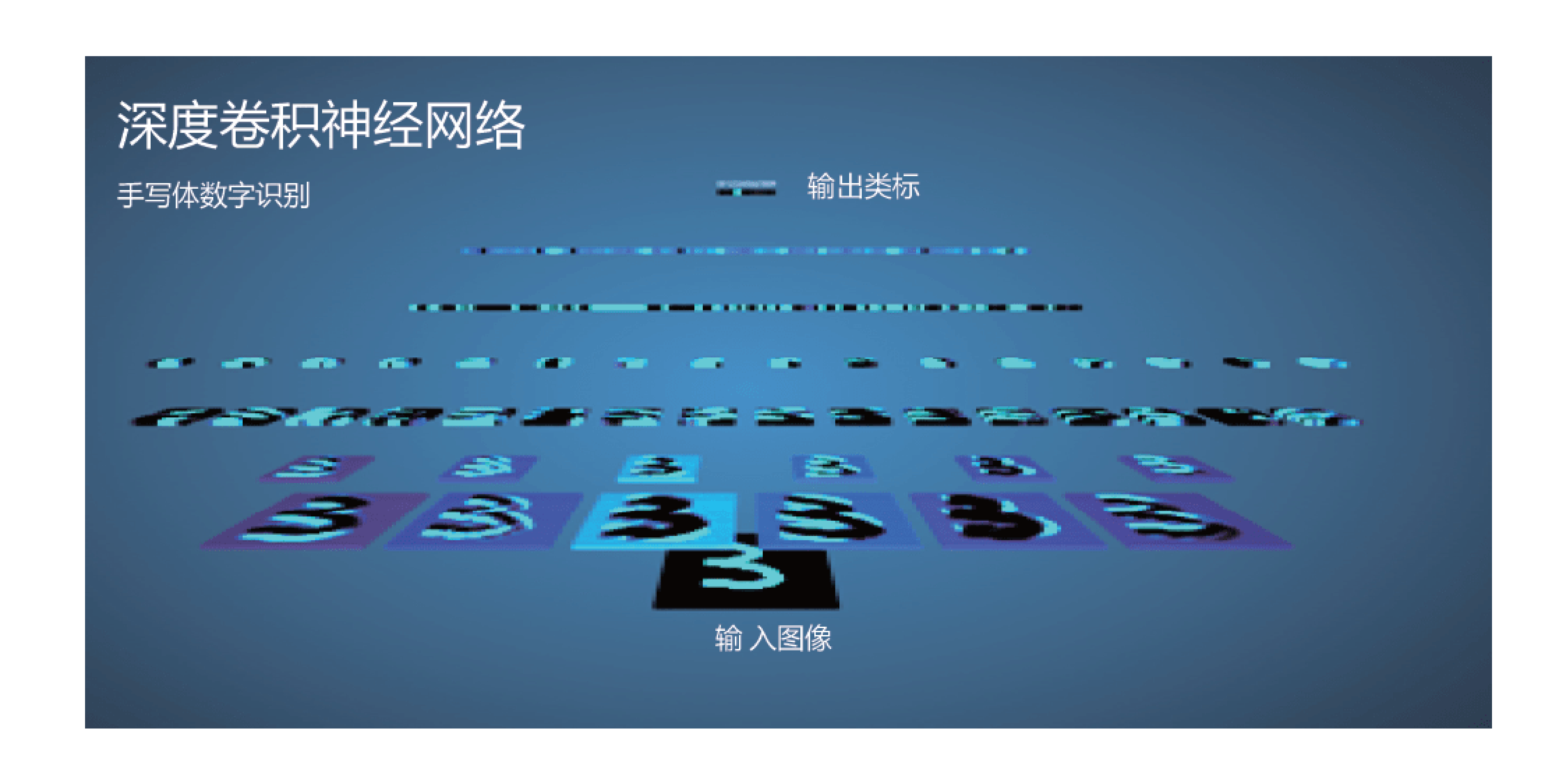
[2.36] Tetko I V，Kovalishyn V V，Luik A I，et al. Variable Selection in the Cascade-Correlation Learning Architecture[M]. Molecular Modeling and Prediction of Bioactivity. Springer US，2000：472-473.

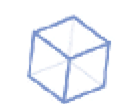
第3章



深度卷积神经网络

CHAPTER 3





3.1 卷积神经网络的生物机理及数学刻画

卷积神经网络是一种特殊的深度前馈神经网络，为了避免层级之间全连接造成的参数冗余，而导致网络模型的训练依赖相当参数个数的数据量；它的设计选择局部连接，符合生物神经元的稀疏响应特性（层级之间的稀疏连接，例如在生物视觉神经系统中，神经元的感受野，即接受区域，具有局部响应特性，只有某个局部区域内的刺激才能够激活该神经元），这样便可以大大降低网络模型的参数规模，相对而言，对训练数据量的依赖性降低。

3.1.1 生物机理

从生物神经研究的角度来看卷积神经网络的发展可知，1960s 年代 Hubel 和 Wiesel 等人通过对猫的视觉皮层细胞的研究，提出了感受野（指听觉系统、视觉系统和感觉系统等中枢神经元的一些性质）这个概念；之后到了 20 世纪 80 年代，Fukushima 在感受野概念的基础上提出了神经认知机的概念，可以看作是卷积神经网络的首次实现，神经认知机将一个视觉模式分解成许多子模式（特征），然后进入分层递阶式相连的特征平面进行处理，它试图将视觉系统模型化，使其能够在即使物体有位移或轻微变形的时候，也能完成识别。

众所周知，视觉皮层（见图 3.1）中的初级视觉皮层（即 V1 区）和中级视觉皮层（即 V2 区）上的细胞可以分为简单细胞和复杂细胞，其中简单细胞的最大程度响应来自感受野范围内的边缘刺激模式，即感受野较小，呈狭长形，用小光点可以测定，对大面积的弥散光无反应，而对处于拮抗区边缘一定方位和一定宽度的条形刺激有强烈的反应，因此比较适合于检测具有明暗对比的直边，对边缘的位置和方位有严格的选择性；而复杂细胞有更大的接受域，它对来自确切位置的刺激具有局部不变性，同时对感受野中的位置无严格要求。（本段文献参考 http://www.lmbe.seu.edu.cn/biology/bess/biology/chapt16/16-2-5.htm）

不论是视觉皮层中的腹侧视觉通路（回答输入是什么，即 What 功能）还是背侧视觉通路（回答输入在场景中的哪个位置，即 Where 功能），对信息的处理都经过初级视觉皮层和中级视觉皮层，其对应的生物响应特性包含：局部感受野特性（空间局部性、空间方向性、信息选择性）；灵长目动物视觉皮层和猫视觉皮层的电生理实验报告和一些相关模型的研究结果都说明了视觉皮层复杂刺激的表达是采用稀疏编码原则的，例如负责视觉感知的视网膜和外侧膝状体的神经细胞只有 100 万个左右（输入神经元个数），而初级视觉皮层 V1 区第四层有 5000 万个（输出神经元个数），但并不是都对前者响应，1996 年，加州大学伯克利分校的 Olshausen 等在 *Nature* 杂志发表论文指出：自然图像经过稀疏编码后得到的基函数类似 V1 区简单细胞感受野的反应特性；所以层与层之间权值的连接为全连接并不总是合理的。

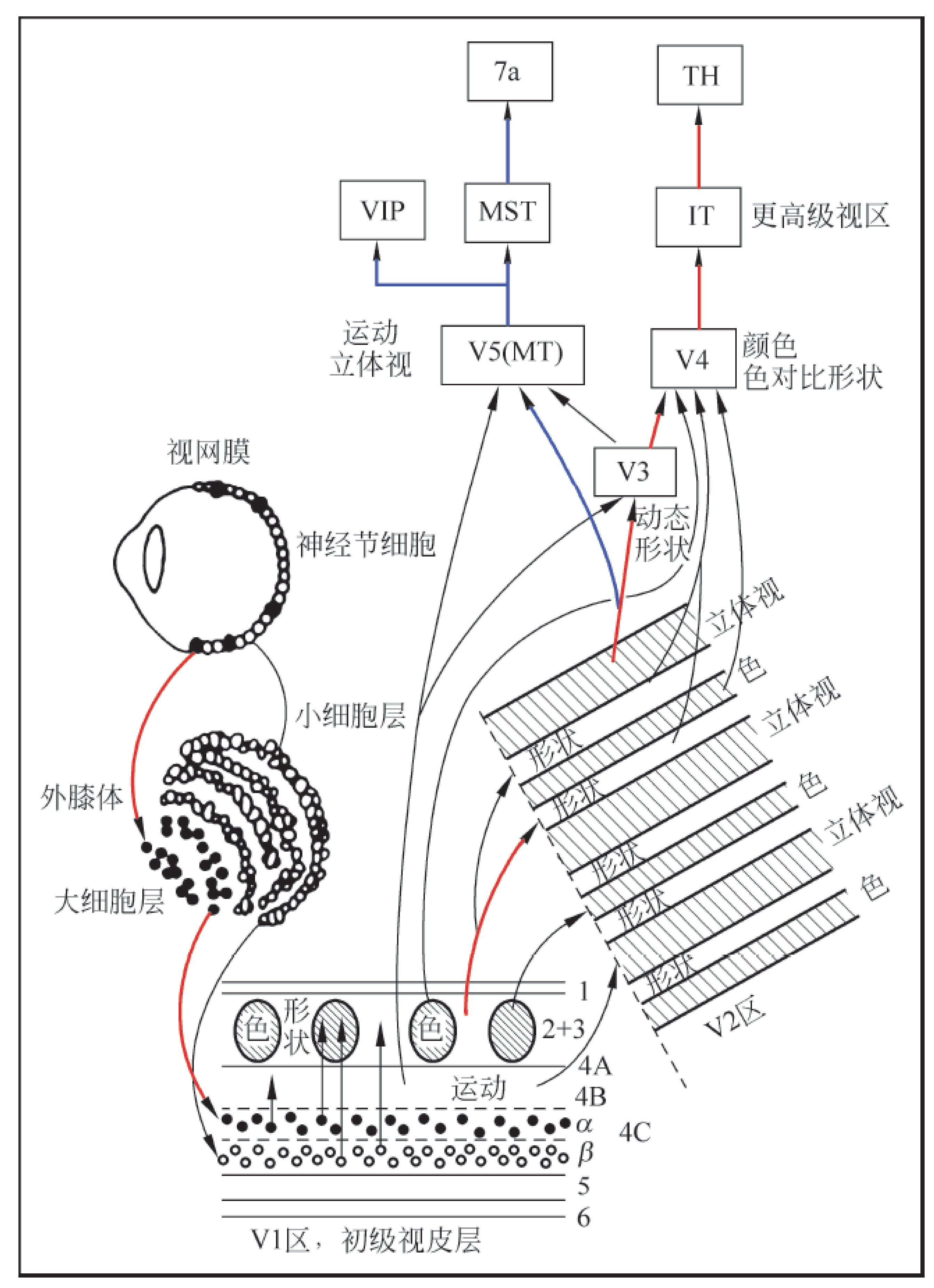


图 3.1　视觉皮层

3.1.2　卷积流的数学刻画

卷积神经网络的基础模块为卷积流，包括卷积（用于维数拓展）、非线性（稀疏性、饱和、侧抑制）、池化（空间或特征类型的聚合）和批量归一化（优化操作，目的是为了加快训练过程中的收敛速度，同时避免陷入局部最优）等四种操作，下面详细讲解这四种操作。

1. 卷积：利用卷积核对输入图片进行处理，可学习到鲁棒性较高的特征

数学中，卷积是一种重要的线性运算；数字信号处理中常用的卷积类型包括三种，即 Full 卷积、Same 卷积和 Valid 卷积。下面假设输入信号为一维信号，即 $\boldsymbol{x}\in\mathbb{R}^n$；且滤波器为一维的，即 $\boldsymbol{w}\in\mathbb{R}^m$，则有：

1）Full 卷积

$$\begin{cases} \boldsymbol{y} = \text{conv}(\boldsymbol{x}, \boldsymbol{w}, \text{'full'}) = (y(1), \cdots, y(t), \cdots, y(n+m-1)) \in \mathbb{R}^{n+m-1} \\ y(t) = \sum_{i=1}^{m} x(t-i+1) \cdot \boldsymbol{w}(i) \end{cases} \tag{3.1}$$

其中 $t=1,2,\cdots,n+m-1$。

2）Same 卷积

$$\boldsymbol{y} = \text{conv}(\boldsymbol{x}, \boldsymbol{w}, \text{'same'}) = \text{center}(\text{conv}(\boldsymbol{x}, \boldsymbol{w}, \text{'full'}), n) \in \mathbb{R}^{n} \tag{3.2}$$

其返回的结果为 Full 卷积中与输入信号 $\boldsymbol{x} \in \mathbb{R}^n$ 尺寸相同的中心部分。

3）Valid 卷积

$$\begin{cases} \boldsymbol{y} = \text{conv}(\boldsymbol{x}, \boldsymbol{w}, \text{'valid'}) = (y(1), \cdots, y(t), \cdots y(n-m+1)) \in \mathbb{R}^{n-m+1} \\ y(t) = \sum_{i=1}^{m} x(t+i-1)\boldsymbol{w}(i) \end{cases} \tag{3.3}$$

其中的 $t=1,2,\cdots,n-m+1$，需要注意 $n>m$。

注意：除了特别声明外，卷积流中常用的是 Valid 卷积。另外，容易将上面一维的卷积操作扩展至二维的操作场景，不再赘述；为了更为直观地说明 Valid 卷积，给出如图 3.2 所示的图示。

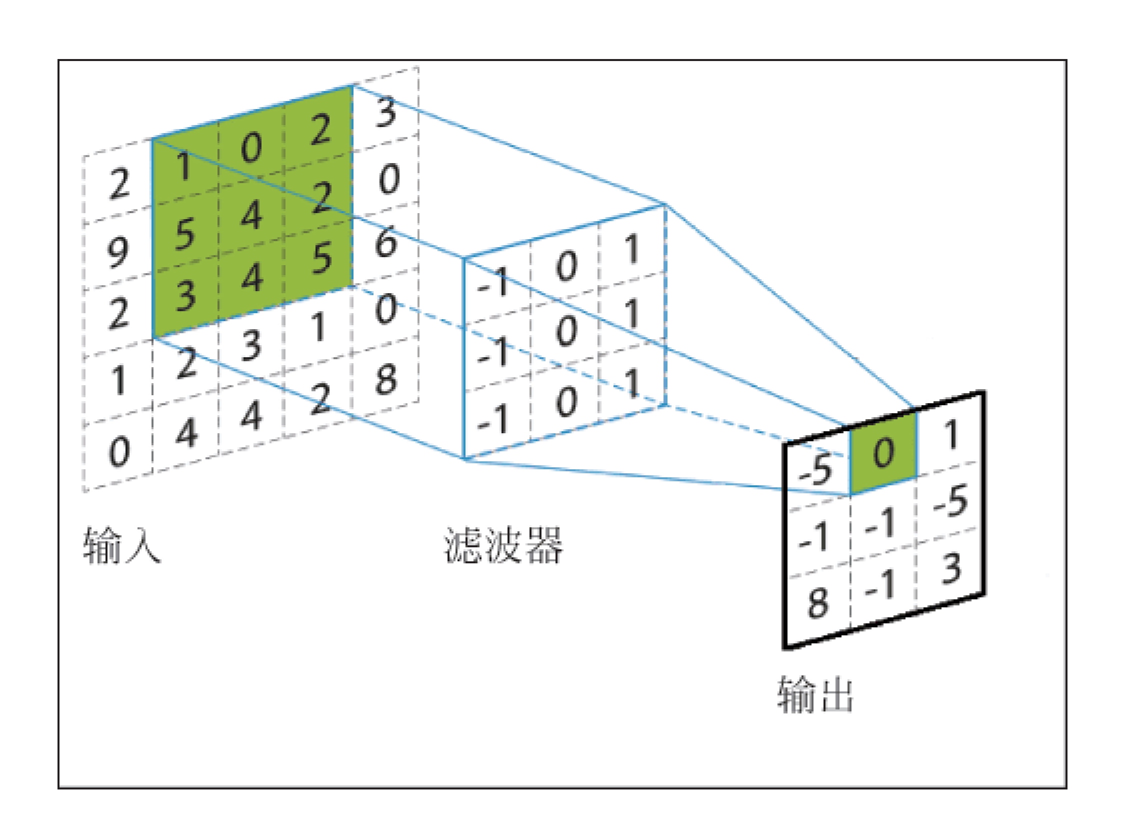


图 3.2　二维 Valid 卷积操作

另外，需要注意的是：深度学习平台 Caffe 中常用的卷积操作包含两个参数即 Stride 和 Zero Padding，其中 Stride 指的是窗口从当前位置到下一个位置，跳过的中间数据个数；例如图像从输入到卷积层的情况，窗口的初始位置在第 1 个像素，第二个位置在第 2 个像素，那么 stride$=2-1=1$；Zero Padding 是指将原始数据的周边补上 0 值的圈数。通常在计算过程中，若输入信号为 $\boldsymbol{x} \in \mathbb{R}^{n\times m}$，卷积核（即滤波器）尺寸大小为 $\boldsymbol{w} \in \mathbb{R}^{s\times k}$，利用 Valid 卷积，同时结合 Stride 和 Zero Padding 得到的输出信号的大小为：

$$\begin{cases} \boldsymbol{y} = \boldsymbol{x} * \boldsymbol{w} \in \mathbb{R}^{u \times v} \\ u = \left\lfloor \dfrac{n - s + 2 \cdot \text{ZeroPadding}}{\text{Stride}} \right\rfloor + 1 \\ v = \left\lfloor \dfrac{m - k + 2 \cdot \text{ZeroPadding}}{\text{Stride}} \right\rfloor + 1 \end{cases} \tag{3.4}$$

其中'$\lfloor \cdot \rfloor$'操作为向下取整。

卷积操作的核心是：可以约减不必要的权值连接，引入稀疏或局部连接，带来的权值共享策略大大地减少参数量相对地提升了数据量，从而可以避免过拟合现象的发生；另外，由于卷积操作具有平移不变性，使得学到的特征具有拓扑对应性、鲁棒性的特性，如图3.3所示，我们分别给出全连接、局部连接和权值共享时所对应的参数，其中权值共享是指相邻神经元的活性相似，从而共享相同的权值参数。

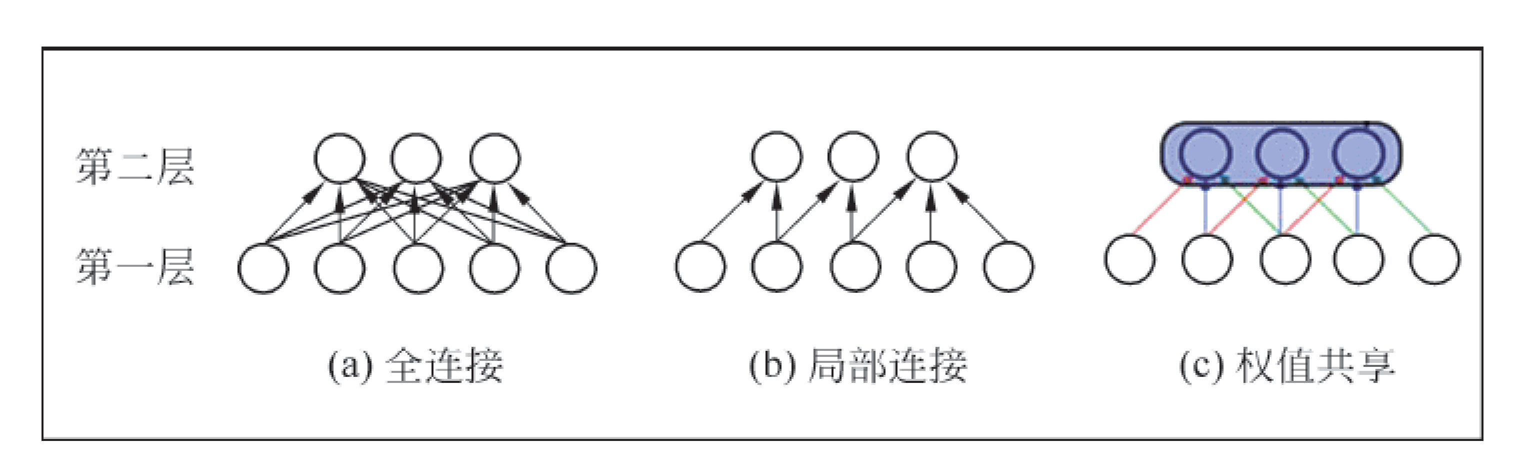


图3.3 连接类型

可以得到全连接(权值连接，不含偏置)的参数为18个，局部连接为7个，权值共享的参数为3个(即黄绿蓝线共用)。

2. 池化：降采样操作，即在一个小区域内，采取一个特定的值作为输出值

本质上，池化操作执行空间或特征类型的聚合，降低空间维度，其主要意义是：减少计算量，刻画平移不变特性；约减下一层的输入维度(核心是对应的下一层级的参数有效地降低)，有效控制过拟合风险。池化的操作方式有多种形式，例如最大池化、平均池化、范数池化和对数概率池化等，常用的池化方式为最大池化(一种非线性下采样的方式)，见图3.4。

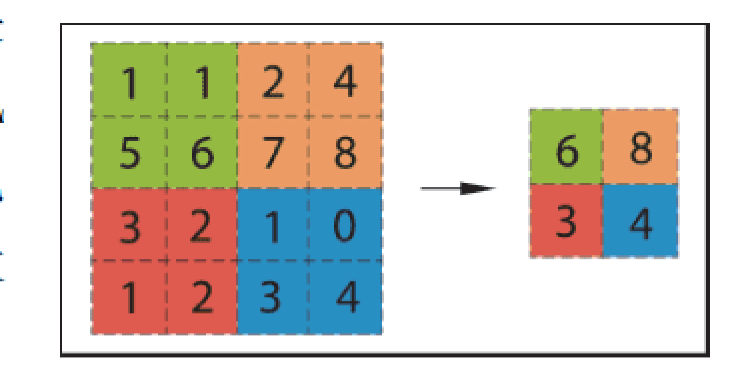


图3.4 最大池化

注意图3.4中是无重叠的最大池化，池化半径为2；在深度学习平台上，除了池化半径以外，还有Stride参数，与卷积阶段的意义相同。

除了上面所举的池化方式外，还有空域金字塔池化方式，它是一种多尺度的池化方式，可以获取输入(指卷积后的特征映射图)的多尺度信息；另外，空域金字塔池化可以把任何尺度的图像的卷积特征转化成相同维度，这不仅可以让卷积神经网络处理任意尺度的图像，还能避免cropping和warping操作所导致的一些信息丢失，具有非常重要的意义。下面利用图简述空域金字塔池化。

需要注意的是，这种空域金字塔池化方式尽可能在最后的卷积流中使用，避免之前的拉伸或向量化所带来的信息丢失。图3.5中卷积层指(可能经过若干层后的)已经得到的最后的特征映射图，共计256个特征映射图；然后以半径分别为1、2、4对这个256维特征映射图进行处理，例如半径为1时，每一个特征映射图(所有元素取最大)得到一个一维的特征，共计256个特征映射图，所以得到256维特征；半径为2是指将每一特征映射图分为四部分，所以可以得到四维特征，共计4×256维特征；以此类推。

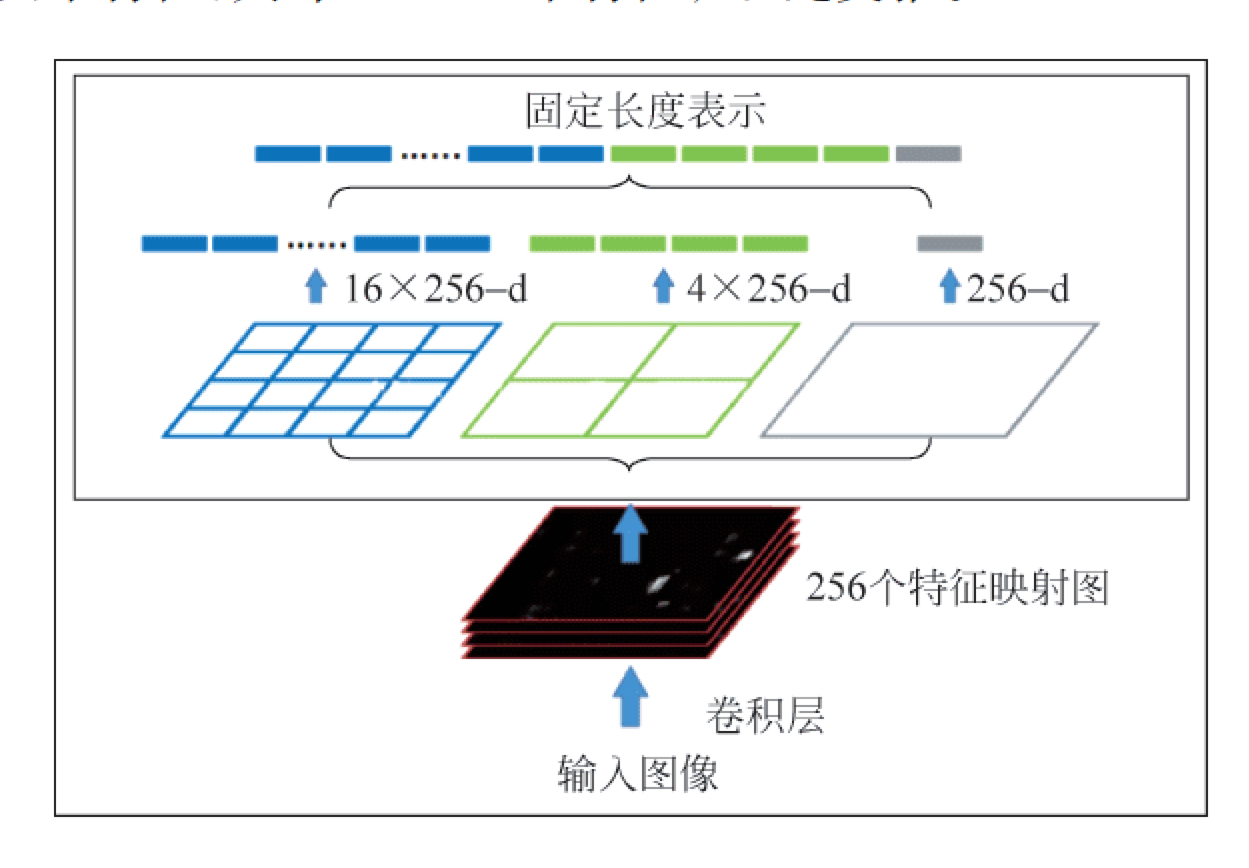


图3.5　空域金字塔池化

3. 激活函数：非线性操作，通过弯曲或扭曲实现表征能力的提升

激活函数的核心是：通过层级(简单)非线性映射的复合使得整个网络的(复杂)非线性刻画能力得到提升，若网络中没有非线性操作，更多的层级组合仍为线性逼近方式，表征或挖掘数据中高层语义特性的能力有限。在应用中，常用的激活函数有：修正线性单元ReLU(加速收敛，内蕴稀疏性)、Softmax(用于最后一层，为计算概率响应)、Softplus函数(ReLU的光滑逼近)、Sigmoid系(传统神经网络的核心所在，包括Logistic-Sigmoid函数和Tanh-Sigmoid函数)；下面我们通过图示来说明这几种激活函数的生物神经特性。

如图3.6所示，从数学上来看，非线性的Sigmoid系对中央区的信号增益较大，对两侧区的信号增益小，在信号的特征空间映射上有很好的效果。但从生物神经科学上来看，中央区酷似神经元的兴奋态，两侧区酷似神经元的抑制态，因而在神经网络学习方面，是将重点特征推向中央区，将非重点特征推向两侧区。随着生物神经科学的发展，2001年神经科学家Dayan、Abott从生物学角度模拟出了脑神经元接收信号更精确的激活模型，如图3.7所示。

与Sigmoid系不同的是：这个生物脑神经元激活函数的主要变化有三点：一是单侧抑制；二是相对宽阔的兴奋边界；三是稀疏激活性(即红框里前端状态完全没有激活)。同年，Charles Dugas等人在做正数回归预测论文中偶然使用了Softplus函数，Softplus的导数便是Logistic-Sigmoid，机器学习领域的Softplus函数和修正线性单元激活函数与神经科

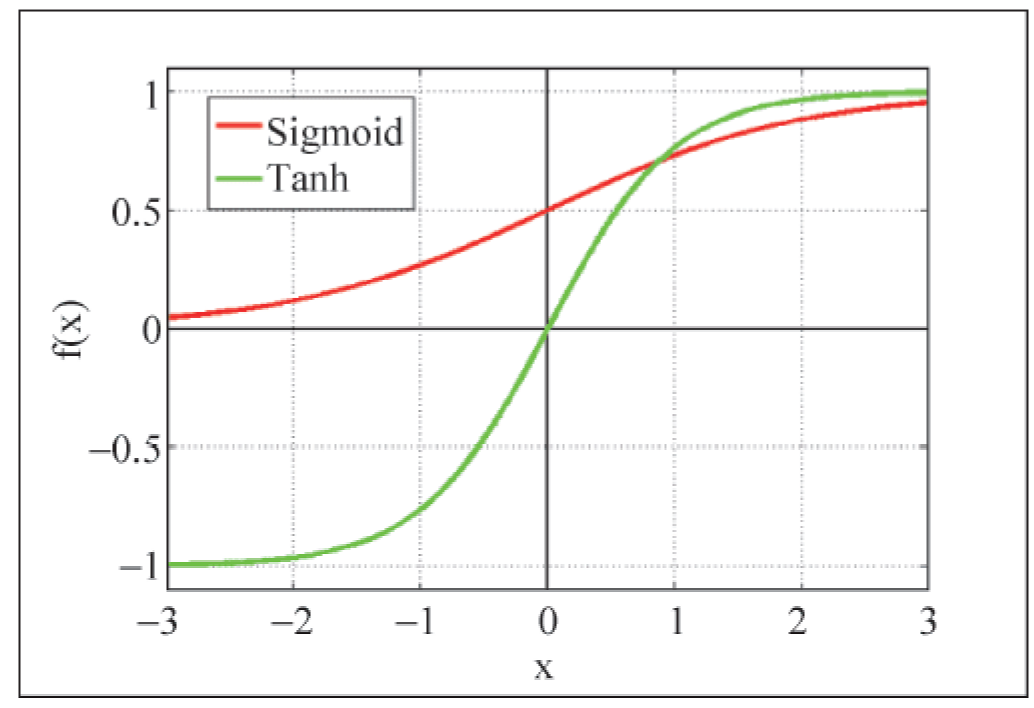


图 3.6　Sigmoid 系——传统神经网络的核心

学领域提出的脑神经元激活频率函数有神似的地方(见图 3.8),这促成了新的激活函数的研究。

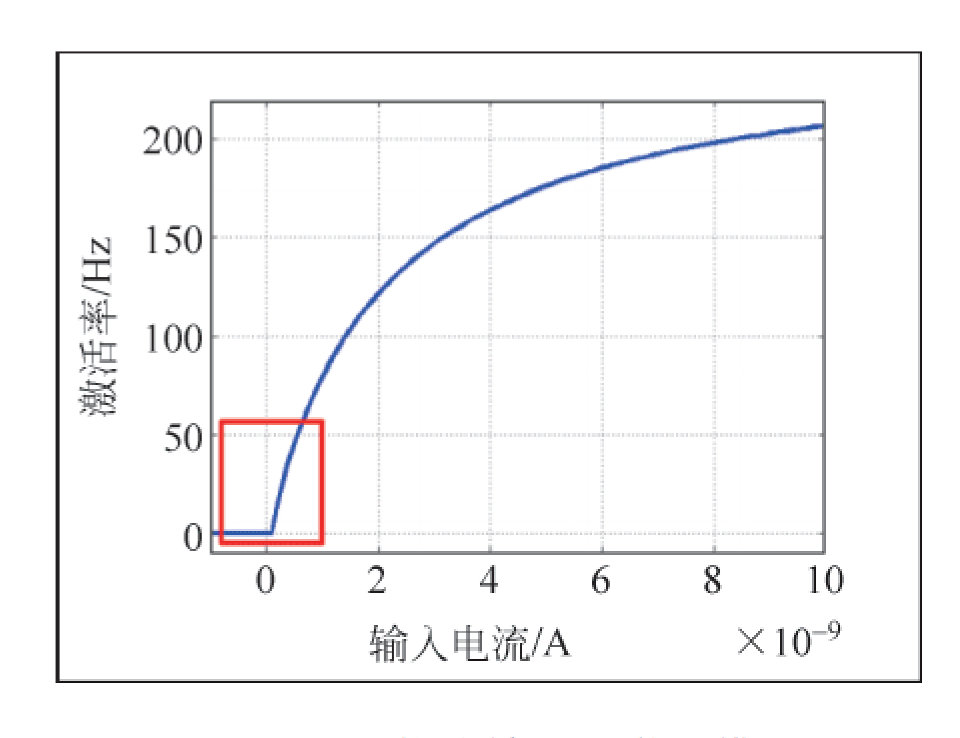


图 3.7　生物脑神经元激活模型

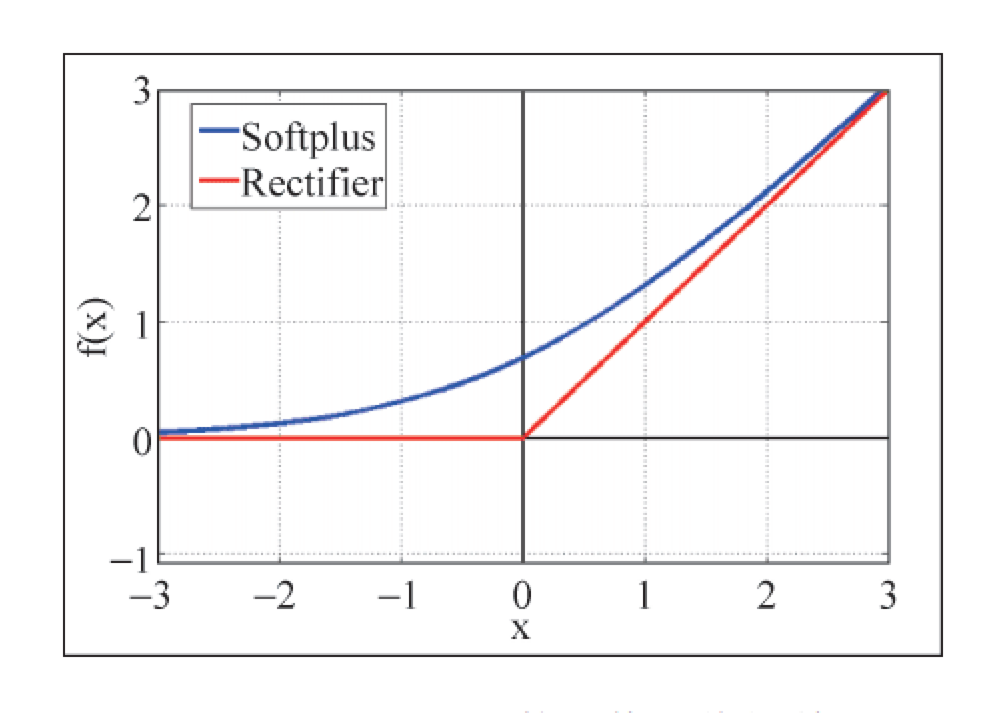


图 3.8　Softplus 函数和修正线性单元

注意:Softplus 函数具有生物脑神经元激活函数的前两点特性,但是不具有稀疏激活特性,而修正线性单元作为 Softplus 函数的逼近,恰好具有这三点特性,并且在深度学习模型中,使用这种简单、速度快的线性激活函数可能更为合适。(此段参考 ReLU 激活函数. http://www.mamicode.com/info-detail-873243.html)

4. 批量归一化:优化操作,减少训练过程中的不稳定性

关于归一化操作,目的是避免随着层级的加深而导致信息的传递呈现逐层衰减的趋势,因为数据范围大的输入在模式分类中的作用可能偏大,而数据范围小的输入作用可能偏小,总之数据范围偏大或偏小,可能导致深度神经网络收敛慢、训练时间长。常用的归一化操作有:L_2范数归一化、Sigmoid 函数归一化(越往两边,区分度越小)等。需要注意的是:卷积神经网络里面有时候会用到各种各样的归一化层,尤其是 2015 年以前的研究,经常能见到它们的身影,但是近些年来的研究表明,这个层级似乎对最后结果的帮助非常小,所以之后

多数时候就摒弃了。

3.2　深度卷积神经网络

下面针对具体的深度卷积神经网络，我们通过模型的架构、训练技巧和模型的优势三个方面来解释其工作原理，以期获得更多关于深度卷积神经网络学习的经验与技巧。

3.2.1　典型网络模型与框架

1. 分类网络——LeNet5

首先给出网络模型(图 3.9)，它是一个非常成功的深度卷积神经网络模型，主要用于手写体数字的识别，应用在银行系统中识别支票上的数字等场景。

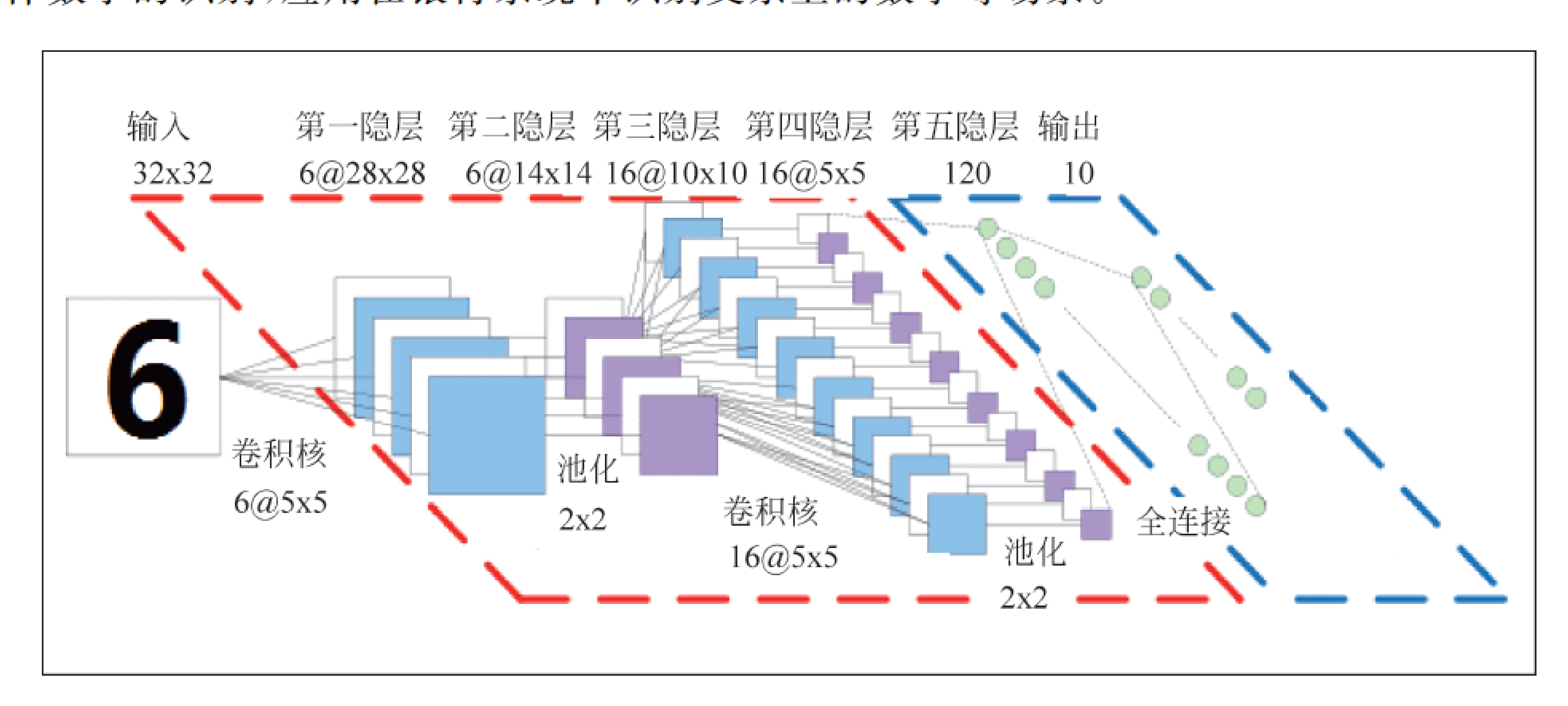


图 3.9　LeNet5 网络的结构

(1) 数据：手写体数据集，分为训练集(共计 10 类，60000 幅)与测试集(共计 10 类，10000 幅)，其中训练与测试集分别记为

$$\begin{cases} \{\boldsymbol{x}_n^{\mathrm{TR}}, y_n^{\mathrm{TR}}\}_{n=1}^{N_{\mathrm{TR}}} \\ \{\boldsymbol{x}_n^{\mathrm{TE}}, y_n^{\mathrm{TE}}\}_{n=1}^{N_{\mathrm{TE}}} \end{cases} \tag{3.5}$$

其中，TR 表示训练，TE 表示测试，输入为 $\boldsymbol{x}_n \in \mathbb{R}^{32\times 32}$，输出为 $y_n \in [0,1,2,\cdots,9]$。

(2) 模型：输入与输出之间的关系如图 3.10 所示，其中左侧虚线框表示特征学习，右侧虚线框表示分类器设计；具体的公式为：

$$\begin{cases} X = \varphi(\boldsymbol{x}, \boldsymbol{W}, b) \\ Y = \mathrm{softmax}(X, \theta) \end{cases} \tag{3.6}$$

其中，X 为输入信号 x 的抽象特征或层次表示特征，参数分为卷积核和偏置：



$$
\begin{cases}
\boldsymbol{W} = [\boldsymbol{W}^1 \in \mathbb{R}^{6@1\times5\times5}, \boldsymbol{W}^2 \in \mathbb{R}^{16@6\times5\times5}, \boldsymbol{W}^3 \in \mathbb{R}^{120@16\times5\times5}] \\
\boldsymbol{b} = [\boldsymbol{b}^1 \in \mathbb{R}^6, \boldsymbol{b}^2 \in \mathbb{R}^{16}, \boldsymbol{b}^3 \in \mathbb{R}^{120}]
\end{cases}
\tag{3.7}
$$

注意：第四隐层与第五隐层之间的全连接理解有两种方式，一是利用卷积的形式获取（深度平台 Caffe 中使用），二是将第四隐层拉伸或向量化（Matlab 下的 Deep Learning Toolbox 中使用），再与第五隐层全连接；这里采用第一种方式。另外，隐层（池化）与隐层（卷积）之间的特征映射图通常需要建立连接表来刻画相应的关系，例如第二隐层与第三隐层，第四隐层与第五隐层等，图 3.10 给出第二隐层与第三隐层之间特征映射图的连接关系。

第二隐层特征映射图编号 \ 第三个隐层特征映射图编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	c				c			c		c	c	c	c		c	c
2	c	c				c		c	c		c	c	c			c
3	c	c	c			c	c	c	c			c		c	c	c
4		c	c	c		c	c	c	c	c			c	c	c	c
5			c	c	c	c	c		c	c	c		c	c		c
6				c	c		c			c	c	c		c	c	c

图 3.10　第二隐层与第三隐层之间的连接关系

字母 c 表示相连，未写出的表示不连接；例如第三隐层的第 1 张特征映射图与第二隐层的第 1，2，3 特征映射图有关系，即有：

$$
\boldsymbol{h}_1^3 = \varphi^2\left(\sum_{j\in I_1} C_{1,j} \cdot (\boldsymbol{w}_{j,1}^2 * \boldsymbol{h}_j^2) + b_1^2\right) \in \mathbb{R}^{10\times10} \tag{3.8}
$$

其中 $\boldsymbol{h}_1^3$ 为第三隐层中的第 1 张特征映射图，这里 $\varphi(t)=t$，b_1^2 为偏置，$C_{1,j}$ 为连接指示集，连接时其值为 1，否则为 0；并且 $I_1=[1,2,3]$ 为关系指示集。若没有连接表，则默认为全连接，即图 3.10 中全部为 c。$\boldsymbol{w}_{j,1}^2 \in \mathbb{R}^{5\times5}$ 为卷积核 $\boldsymbol{w}^2$ 所对应第三隐层第一个特征映射图与第二隐层第 j 个特征映射图之间的滤波器。

对于分类器设计阶段，其参数为：

$$
\begin{cases}
Y(k) = P(y = k \mid X, \theta_k) = \dfrac{e^{X^{\mathrm{T}}\cdot\theta_k}}{\sum\limits_{s=0}^{9} e^{X^{\mathrm{T}}\cdot\theta_s}} \in \mathbb{R} \\
\theta = [\theta_0, \theta_1, \cdots, \theta_9]
\end{cases}
\tag{3.9}
$$

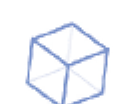
其中 $k=0,1,2,\cdots,9$；最后输出的类标为：

$$
y = \arg\max_k \{Y(k)\} \tag{3.10}
$$

（3）优化目标函数：在训练数据集上，利用交叉熵来构造目标函数为：

$$
\begin{aligned}
\min_{\{\boldsymbol{W},b;\theta\}} J(\boldsymbol{W},b;\theta) =& -\frac{1}{N_{\mathrm{TR}}}\sum_{n=1}^{N_{\mathrm{TR}}}\sum_{k=0}^{9}\delta(y_n^{\mathrm{TR}} = k)\cdot\log(Y_n^{\mathrm{TR}}(k)) \\
&+ \lambda_1 R(\boldsymbol{W}) + \lambda_2 R(\theta)
\end{aligned}
\tag{3.11}
$$

其中后两项为正则项，另外具体的符号表示为：



$$\begin{cases} Y_n^{\mathrm{TR}}(k) = \mathrm{softmax}(X_n^{\mathrm{TR}}, \theta) = \mathrm{softmax}(\varphi(x_n^{\mathrm{TR}}; \boldsymbol{W}, b), \theta) \\ R(\boldsymbol{W}) = \sum_{l=1}^{3} \| \boldsymbol{W}^l \|_F^2 \\ R(\theta) = \sum_{k=0}^{9} \| \theta_k \|_F^2 \end{cases} \tag{3.12}$$

（4）求解：利用梯度下降法来实现优化目标函数中参数的学习，由于目标函数随着层级的加深，而导致非凸优化问题（在参数所构成的超平面中，大量地存在着鞍点与局部极值点），求解前需给定较好的参数初始值。与深度前馈神经网络中所使用的反向传播算法在计算中有所不同的是：利用误差反向传播时，需要考虑池化隐层向卷积隐层的误差传播公式，以及卷积隐层向池化隐层传播的误差公式。首先更新参数的优化公式为：

$$\begin{cases} \boldsymbol{W}^{(k)} = \boldsymbol{W}^{(k-1)} - \alpha \cdot \dfrac{\partial J(\boldsymbol{W}, b; \theta)}{\partial \boldsymbol{W}} \Big|_{\boldsymbol{W}=\boldsymbol{W}^{(k-1)}} \\ b^{(k)} = b^{(k-1)} - \alpha \cdot \dfrac{\partial J(\boldsymbol{W}, b; \theta)}{\partial b} \Big|_{b=b^{(k-1)}} \\ \theta^{(k)} = \theta^{(k-1)} - \alpha \cdot \dfrac{\partial J(\boldsymbol{W}, b; \theta)}{\partial \theta} \Big|_{\theta=\theta^{(k-1)}} \end{cases} \tag{3.13}$$

式(3.13)中的参数 θ 由于是分类器设计阶段，采用之前的层级全连接结构，所以与前馈神经网络中的求导是一致的，这里不再赘述；下面主要关心目标函数关于隐层偏导的求解（误差传播）方式。

为了方便说明，我们完整地引入以下符号系统：

$$\left.\begin{aligned} \boldsymbol{h}_l^1 &= \varphi^1(\boldsymbol{W}_l^1 * x + b_l^1) \in \mathbb{R}^{28\times 28} \\ \boldsymbol{h}_l^2 &= \mathrm{Maxpooling}(\boldsymbol{h}_l^1, r^2) \in \mathbb{R}^{14\times 14} \end{aligned}\right\} \longrightarrow l = 1,2,\cdots,6; \quad \varphi^1(t) = t$$

$$\left.\begin{aligned} \boldsymbol{h}_s^3 &= \varphi^2\left(\sum_{l\in I_s} \boldsymbol{W}_{l,s}^2 * \boldsymbol{h}_l^2 + b_s^2\right) \in \mathbb{R}^{10\times 10} \\ \boldsymbol{h}_s^4 &= \mathrm{Maxpooling}(\boldsymbol{h}_s^3, r^4) \in \mathbb{R}^{5\times 5} \end{aligned}\right\} \longrightarrow s = 1,2,\cdots,16; \quad \varphi^2(t) = t$$

$$\left.\begin{aligned} h_q^5 &= \varphi^3\left(\sum_{s\in I_q} \boldsymbol{W}_{s,q}^3 * \boldsymbol{h}_s^4 + b_q^3\right) \in \mathbb{R}^{1\times 1} \\ \boldsymbol{h}^5 &= (h_1^5, h_2^5, \cdots, h_{120}^5) \in \mathbb{R}^{120\times 1} \\ \boldsymbol{h}^6 &= \mathrm{softmax}(\boldsymbol{h}^5, \theta) \in \mathbb{R}^{10\times 1} \end{aligned}\right\} \longrightarrow q = 1,2,\cdots,120; \quad \varphi^3(t) = \max(0,t)$$

目标函数关于第五隐层的误差传播梯度为：

$$\boldsymbol{\delta}^{(r)} = \frac{\partial J(\boldsymbol{W}, b; \theta)}{\partial \boldsymbol{h}^r} \tag{3.14}$$

其中 $r=1,2,3,4,5$；相应的参数更新公式为：

$$\begin{cases} \dfrac{\partial J(\boldsymbol{W}, b; \theta)}{\partial \boldsymbol{W}^s} = \dfrac{\partial \boldsymbol{h}^{2\cdot s-1}}{\partial \boldsymbol{W}^s} \dfrac{\partial J(\boldsymbol{W}, b; \theta)}{\partial \boldsymbol{h}^{2\cdot s-1}} = \dfrac{\partial \boldsymbol{h}^{2\cdot s-1}}{\partial \boldsymbol{W}^s} \cdot \delta^{(2\cdot s-1)} \\ \dfrac{\partial J(\boldsymbol{W}, b; \theta)}{\partial b^s} = \dfrac{\partial \boldsymbol{h}^{2\cdot s-1}}{\partial b^s} \dfrac{\partial J(\boldsymbol{W}, b; \theta)}{\partial \boldsymbol{h}^{2\cdot s-1}} = \dfrac{\partial \boldsymbol{h}^{2\cdot s-1}}{\partial b^s} \cdot \boldsymbol{\delta}^{(2\cdot s-1)} \end{cases} \tag{3.15}$$



其中 $s=1,2,3$；其中求解的核心是式(3.14)。

2. 目标检测网络——faster R-CNN

如图 3.11 所示给出网络模型,它也是一个非常成功的深度卷积神经网络模型,主要用于目标检测,通常简记为 faster R-CNN；注意该任务与以往的分类识别任务不同的是：需要在场景中实现目标定位(即回答 Where)。首先,经典的基于候选区域的卷积神经网络(R-CNN)是通过选择搜索方法实现场景中目标候选区域的选取(目标候选框选取,大致每一幅场景中选择近两千个候选区域；再根据相互交叠的面积选择合适的阈值对候选区域再选择)；进一步,对候选区域实现卷积神经网络的特征提取与支撑矢量机实现分类。在其基础上,提出改进的模型有 fast R-CNN(候选区的选择仍沿用 R-CNN 中的选择搜索方法,但识别阶段利用感兴趣区域 RoI 池化层来实现网络模型任意的输入与固定的输出,分类器为 Softmax 函数)和 faster R-CNN(核心理念是端到端的设计模式,要求场景中目标区域的定位与识别同时输出,创新工作在于提出了区域生成网络——用于候选区的选择与定位)。这三种方法的对比见表 3.1。

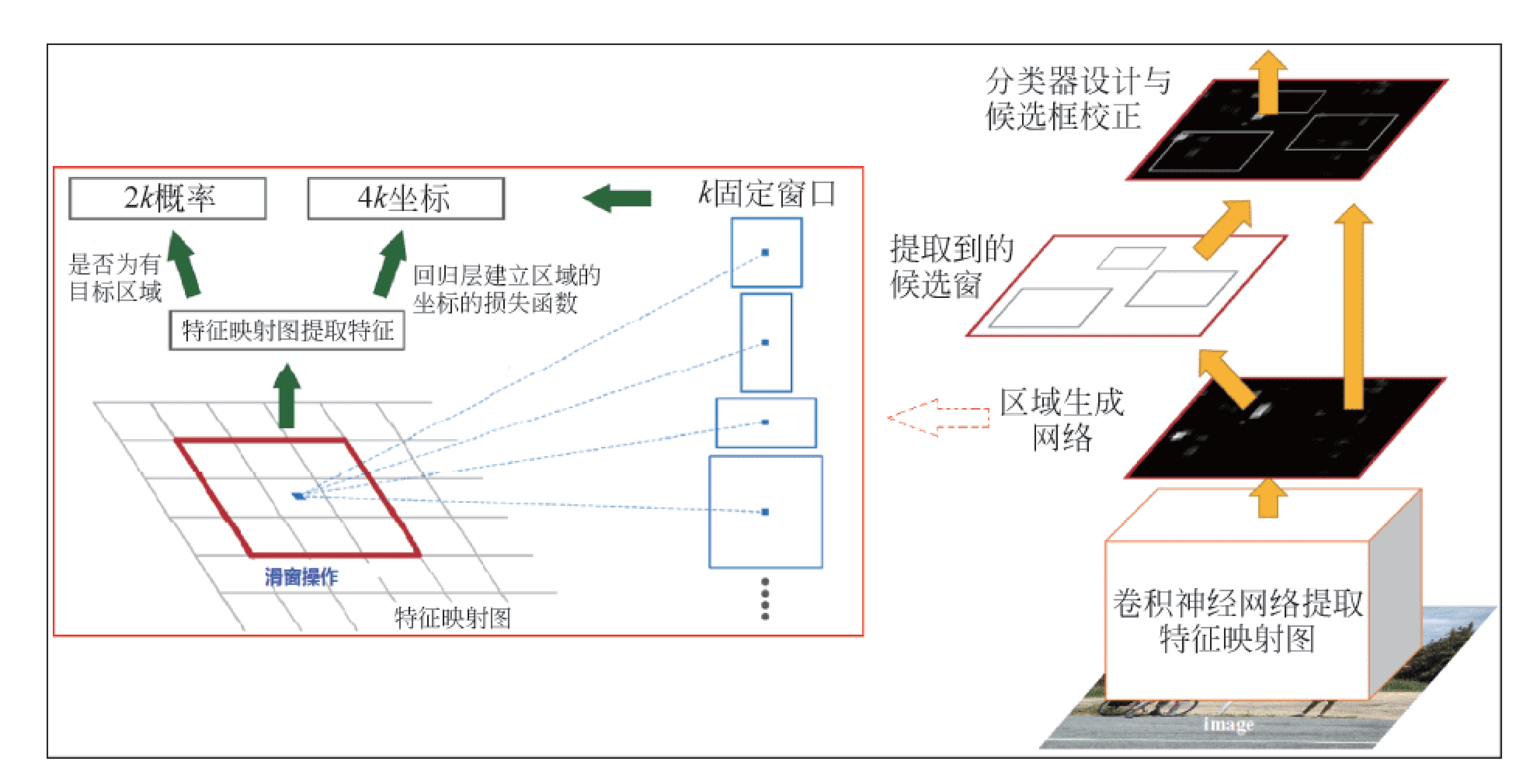


图 3.11　faster R-CNN 网络的结构

表 3.1　经典目标检测算法的对比

项目＼方法	R-CNN	fast R-CNN	faster R-CNN
候选区域生成 Where	基于选择搜索方法的输入场景中目标候选区域的提取	基于选择搜索方法的输入场景中目标候选区域的提取	基于 fast R-CNN 中目标识别与辅助任务的统一框架,引入"注意"机制——区域生成网络,使得对于目标检测任务形成一个统一的网络框架
目标识别 What	深度卷积神经网络用于特征提取＋SVM 分类器的结合	利用端到端的方式实现目标识别,同时实现候选区域位置精修,核心提出 ROI 池化层	
辅助任务	候选区域位置精修		

其次,需要注意的是:faster R-CNN 引入了"注意"机制——区域生成网络(用于候选区域的生成),代替了 fast R-CNN 中的选择搜索方法,所以 faster R-CNN 可以简单地看作是区域生成网络(Where)与 fast R-CNN(What)的结合,输入与输出之间的处理流程见图 3.12。

备注:关于目标检测网络参考链接 http://blog.csdn.net/column/details/ym-alanyannick.html.

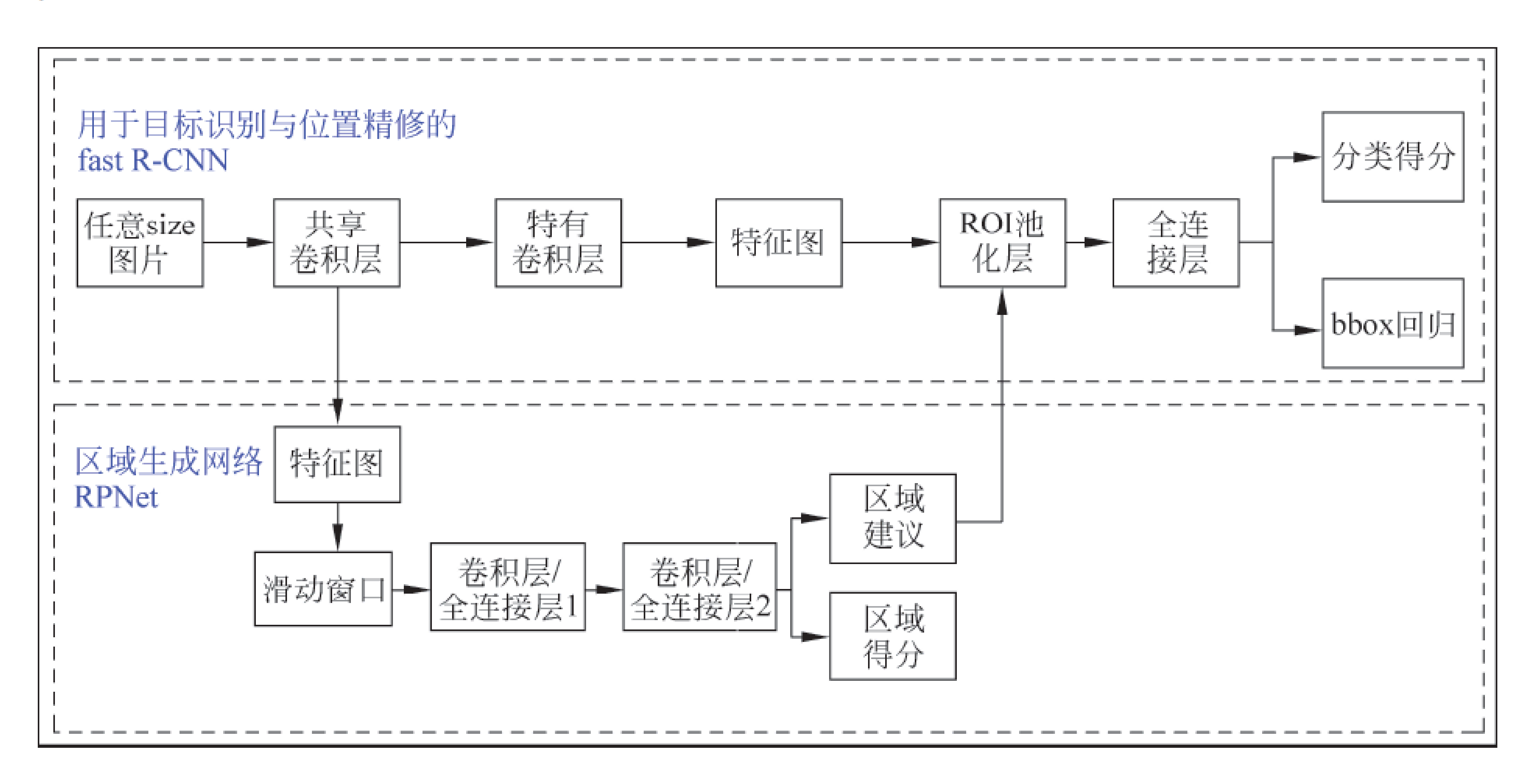


图 3.12 Faster R-CNN 网络处理流程

(1) 数据:数据仍分为两部分,训练数据集和测试数据集,这里不再给出相应的数据量,仅给出输入与输出数据的解释:

$$\{\boldsymbol{x}_n, y_n\}_{n=1}^{N} \tag{3.16}$$

其中 $\boldsymbol{x}_n \in \mathbb{R}^{n\times m}$ 为输入,即场景;y_n 为期望输出,包括两部分,一是场景中所有目标的位置(利用目标区域左上角的位置,以及宽与高来定位),二是目标区域所对应的物体类别,可写为:

$$\begin{cases} y_n = [P_{x_n-\text{target}}^{(s)} \in \mathbb{R}^4, \text{Lable}_{x_n-\text{target}}^{(s)} \in [1,2,\cdots,C]] \\ s = 0,1,2,\cdots,S \end{cases} \tag{3.17}$$

其中的符号解释为:这一幅场景 $\boldsymbol{x}_n$ 中有 S 个目标,其中每一个目标所对应的坐标 $P_{x_n-\text{target}}^{(s)}$ 和类标 $\text{Lable}_{x_n-\text{target}}^{(s)}$,注意识别场景中的目标种类共有 C 类。另外式(3.16)中的输入场景个数为 N,需要注意的是:每一幅场景有可能有目标,也有可能没有目标(即当 $S=0$ 时)。

注意:常用的数据集为网络数据集 ImageNet、Pascal VOC2012、医学腹部肿瘤数据 sunny-brack 等。

(2) 模型:输入与输出的处理流程见图 3.12,该框架非常清晰地给出了实现的每一步:

其中共享/特有卷积层后的特征图提取部分可以利用各种经典的卷积神经网络来实现

(注意：将这些经典网络最后的分类部分去掉，或者在某特征映射层后去掉后面的部分；例如 VGG 网络、AlexNet 网络、GoogleNet 网络、ZF 网络等)，不论输入场景为灰度图还是彩色图，处理流程一样，这里不再赘述。

下面通过公式来描述输入与输出之间的关系，分为如下的两个部分。

第一部分：区域生成网络：

$$\begin{cases} \boldsymbol{X}_{\text{Part1}} = \text{ConvNet}^{(\text{Part1})}(\boldsymbol{x}, \theta_{\text{Part1}}) \in \mathbb{R}^{u \times v @ r} \\ \text{RP}_x = [\text{RP}_x^1 \in \mathbb{R}^{2 @ \widetilde{S}}, \text{RP}_x^2 \in \mathbb{R}^{4 @ \widetilde{S}}] = \text{RPNet}(\boldsymbol{X}_{\text{Part1}}(\boldsymbol{x}), \theta_{\text{RP}}) \end{cases} \tag{3.18}$$

第二部分：结合区域生成网络的输出，得到 fast R-CNN 网络的输入与输出之间的关系：

$$\begin{cases} \boldsymbol{X}_{\text{Part1}} = \text{ConvNet}^{(\text{Part1})}(\boldsymbol{x}, \theta_{\text{Part1}}) \in \mathbb{R}^{u \times v @ r} \\ \boldsymbol{X}_{\text{Part2}} = \text{ConvNet}^{(\text{Part2})}(\boldsymbol{X}_{\text{Part1}}, \theta_{\text{Part2}}) \\ \boldsymbol{X}_{\text{Part3}} = \text{ROI}^{(\text{Part3})}(\boldsymbol{X}_{\text{Part2}}, \text{RP}_x) \\ y = [\text{RP}_x^{(\tilde{s})}, \text{Lable}_x^{(\tilde{s})}] = \text{FC}(\boldsymbol{X}_{\text{Part3}}, \theta_{\text{Part3}}, C, \text{Refine}(\text{RP}_x)) \\ \tilde{s} = 0, 1, 2, \cdots, \widetilde{S} \end{cases} \tag{3.19}$$

进一步具体解释第一部分，即公式(3.18)描述如下：

第一步：共享卷积层后的特征图生成。

$$\boldsymbol{X}_{\text{Part1}} = \text{ConvNet}^{(\text{Part1})}(\boldsymbol{x}, \theta_{\text{Part1}}) \in \mathbb{R}^{u \times v @ r}$$

即将输入场景 $\boldsymbol{x} \in \mathbb{R}^{n \times m}$，通过第一部分卷积神经网络实现特征图的提取，这一步输出为 $\boldsymbol{X}_{\text{Part1}} \in \mathbb{R}^{u \times v @ r}$，即有 r 个特征图，每一个尺寸为 $u \times v$；其中待学习的参数记为 θ_{Patr1}。

第二步：利用区域生成网络实现候选目标区域的提取。

关于区域生成网络的训练，需明确该网络将一个图像(任意大小)作为输入，输出候选目标(矩形)区域的集合，并且对每个(矩形)区域给出是否为目标的得分，即如下公式中的 $\boldsymbol{x}$ 为输入图像，RP_x 为输出，其中 $\widetilde{S}$ 为候选目标(矩形)区域的个数：

$$\text{RP}_x = [\text{RP}_x^1 \in \mathbb{R}^{2 @ \widetilde{S}}, \text{RP}_x^2 \in \mathbb{R}^{4 @ \widetilde{S}}] = \text{RPNet}(\boldsymbol{X}_{\text{Part1}}(\boldsymbol{x}), \theta_{\text{RP}})$$

RP_x^1 为每个候选区域判断为目标、背景区域的得分，对应着 RP_x^2 为候选目标区域的位置；待学习的参数为 θ_{RP}；该网络的训练样本集为：

$$\{\boldsymbol{x}_n, \tau_n\}_{n=1}^{N} \tag{3.20}$$

注意区域生成网络的输入与 fast R-CNN 的输入是一致的，但二者的输出却不一样。这里的 τ_n 有两个部分，一是目标区域的得分(即判断矩形区域为目标或背景的分数，二分类)，另一个是目标区域的位置；由于是二分类问题，所以需构建基于特征图 $\boldsymbol{X}_{\text{Part1}}$ 的正负样本集来训练 RPNet 网络中的参数 θ_{RP}；如何构建基于特征图 $\boldsymbol{X}_{\text{Part1}}$ 的正负样本集？对训练集中的每幅输入场景，依据每个标定的真值目标(矩形)区域与候选目标(矩形)区域的重叠比例(IOU)大于 0.7，为正样本；其比例都小于 0.3，记为负样本；其余候选区域舍弃。如何得到候选目标(矩形)区域？利用输入场景与特征图 $\boldsymbol{X}_{\text{Part1}}$ 之间的拓扑结构关系，即输入场景的某

个矩形区域与特征图中的某个矩形区域有着一一对应的比例关系，依据特征图 $\boldsymbol{X}_{\text{Part1}} \in \mathbb{R}^{u \times v @ r}$ 中 $u \times v$ 平面上的每个位置，进行滑窗处理(需给出窗口大小，例如 3×3 等)，同时该位置的 r 维特征来表征该窗口的特性(即特征向量)，为了使得特征图中某一位置的窗口具有多样性，引入窗口的大小和比例这两个参数，得到表征特征图中某一位置的多个矩形区域(不妨记为 k 个矩形区域)，并将这些矩形区域通过拓扑对应关系映射至输入场景中，得到所谓的 Anchor，即候选目标(矩形)区域(若将特征图中所有的位置都遍历一次，则整个候选目标(矩形)区域共计有 $u \times v \times k$)。得到的正负样本集记为：

$$\begin{cases} \{\{p_i^{(n)}, t_i^{(n)}\}_{i=1}^{\kappa_n}\}_{n=1}^{N} \\ \sum_{n=1}^{K} \kappa_n = \kappa \end{cases} \tag{3.21}$$

其中 κ_n 为第 n 幅场景所得到样本集的个数，总的(正负)样本集个数为 κ。对比式(3.20)与式(3.21)，可以知道：

$$\tau_n = \{p_i^{(n)}, t_i^{(n)}\}_{i=1}^{\kappa_n} \tag{3.22}$$

第二部分，即式(3.19)描述如下：

第一步：共享卷积层后的特征图生成，这一步与第一部分区域生成网络的第一步一样，共享计算结果，这里不再赘述。

第二步：特有卷积层后的特征图

$$\boldsymbol{X}_{\text{Part2}} = \text{ConvNet}^{(\text{Part2})}(\boldsymbol{X}_{\text{Part1}}, \theta_{\text{Part2}})$$

这一步主要利用共享卷积层后的特征图 $\boldsymbol{X}_{\text{Part1}}$ 来得到 $\boldsymbol{X}_{\text{Part2}}$，其中待学习的网络参数为 θ_{Part2}。

第三步：感兴趣区域(ROI)的池化层输出

$$\boldsymbol{X}_{\text{Part3}} = \text{ROI}^{(\text{Part3})}(\boldsymbol{X}_{\text{Part2}}, \text{RP}_x)$$

这一步的输入为区域生成网络得到的候选区域 RP_x(具有较高目标区域得分的区域建议)和第二步的输出 $\boldsymbol{X}_{\text{Part2}}$，由于候选区域的尺寸大小不一，为了避免裁剪或缩放所带来的信息损失，引入单层空域塔式池化(SPP)来实现不同尺寸的输入、相同尺寸的输出。

第四步：全连接层后的(预测)的输出

$$y = [\text{RP}_x^{(\tilde{s})}, \text{Lable}_x^{(\tilde{s})}] = \text{FC}(\boldsymbol{X}_{\text{Part3}}, \theta_{\text{Part3}}, C, \text{Refine}(\text{RP}_x))$$

输出的 y 包括两部分，一是目标区域的位置，二是目标区域的类标，其中该层的输入为第三步的输出 $\boldsymbol{X}_{\text{Part3}}$ 以及分类任务中的类别个数 C，同时对每一类所对应的目标区域进行精修的参数 $\text{Refine}(\text{RP}_x, c)$(即目标矩形区域中左上坐标与长、宽、高等滑动的位移)，其中 $c=1,2,\cdots,C$。另外每一幅输入场景，可能对应着目标区域的个数为 $\tilde{s}=0,1,2,\cdots,\tilde{S}$。

(3) 优化目标函数：

接下来，针对模型，待训练的参数包括：

$$\begin{cases} \text{RPNet}: (\theta_{\text{Part1}}, \theta_{\text{RP}}) \\ \text{fast R-CNN}: (\theta_{\text{Part1}}, \theta_{\text{Part2}}, \theta_{\text{Part3}}, \{\text{Refine}(\text{RP}_x, c)\}_{c=1}^{C}) \end{cases} \tag{3.23}$$



其中共享卷积层的计算应包括两条设计通路,下面通过图示(图 3.13)给出优化的策略与待求解的目标函数。

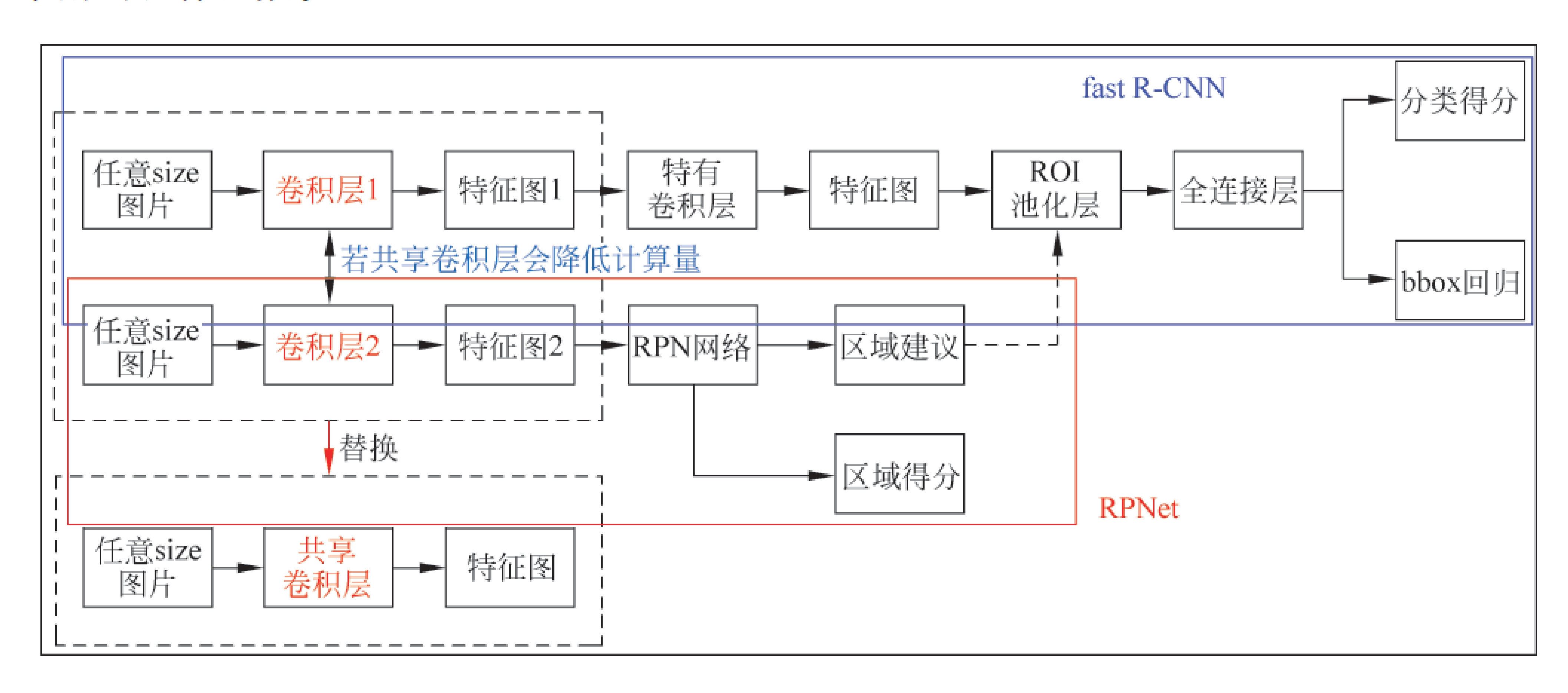


图 3.13　Faster R-CNN 网络优化路径

其中虚线框为共享计算的部分,最上侧实线框为 fast R-CNN 优化的通路(主要完成目标识别与位置精修),中间实线框为 RPNet 优化通路(完成“注意”机制,即候选目标(矩形)区域的生成)。综上所述,优化目标函数有两个。

第一个优化目标函数是 RPNet 优化通路。根据模型中第一部分的描述,优化目标函数为:

$$\min_{\substack{\theta_{\text{part1}}\\ \theta_{\text{RP}}}} J(\theta_{\text{part1}},\theta_{\text{RP}}) = \frac{1}{N}\sum_{n=1}^{N} L(\boldsymbol{x}_n,\tau_n;\theta_{\text{part1}},\theta_{\text{RP}}) + \lambda_1 R(\theta_{\text{part1}}) + \lambda_2 R(\theta_{\text{part2}}) \tag{3.24}$$

式(3.24)由两个部分构成,一部分为经验风险评估损失项,另一部分为正则项,关于正则项的约束,利用富比尼斯范数。接下来,每一幅场景的损失函数为:

$$\begin{cases} L(\boldsymbol{x}_n,\tau_n;\theta_{\text{Part1}},\theta_{\text{RP}}) = \dfrac{1}{N_{\text{cls}}}\displaystyle\sum_{i=1}^{K_n} L_{\text{cls}}(\hat{p}_i^{(n)},p_i^{(n)}) + \dfrac{1}{N_{\text{reg}}}\displaystyle\sum_{j=1}^{K_n} p_j^{(n)} L_{\text{reg}}(\hat{t}_j^{(n)},t_j^{(n)}) \\ L_{\text{cls}}(\hat{p}_i^{(n)},p_i^{(n)}) = -\log[\hat{p}_i^{(n)}(1)\cdot p_i^{(n)}(1) + (1-\hat{p}_i^{(n)}(1))(1-p_i^{(n)}(1))] \in \mathbb{R} \\ L_{\text{reg}}(\hat{t}_j^{(n)},t_j^{(n)}) = R(\hat{t}_j^{(n)} - t_j^{(n)}) = \displaystyle\sum_{p=1}^{4} R(\hat{t}_j^{(n)}(p) - t_j^{(n)}(p)) \in \mathbb{R} \end{cases} \tag{3.25}$$

简单解释如下:该损失函数由两部分构成,一部分为区域得分损失(分类器的设计),另一部分为候选区域的位置(回归器的设计),二者之间相互影响,如回归器中的权重因子 $p_j^{(n)}(1)$,即为将第 n 幅场景中的第 j 个候选区域判断为目标区域的概率,另外式(3.25)中回归器的非线性函数为:

$$R(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{else} \end{cases} \tag{3.26}$$

第二个优化目标函数是 fast R-CNN 优化通路

$$\min_{\theta} J(\theta_{\text{Part1}},\theta_{\text{Part2}},\theta_{\text{Part3}},\{\text{Refine}(\text{RP}_x,c)\}_{c=1}^{C})$$
$$=\frac{1}{N}\sum_{n=1}^{N}[L_{\text{cls}}(\boldsymbol{x}_n,y_n;\theta)+L_{\text{loc}}(\boldsymbol{x}_n,y_n;\text{Refine}(\text{RP}_{x_n}))] \tag{3.27}$$

其中 $\theta=[\theta_{\text{Part1}},\theta_{\text{Part2}},\theta_{\text{Part3}}]$，注意该通路的输出与区域生成网络的输出不同。对于输出的 y_n 可以参考式(3.16)，另外优化目标函数包括两部分，一部分是区域分类损失(这里的分类指的是目标共分为 C 类，计算每一候选区域分到 c 的概率，与区域生成网络中的两分类(目标或背景)任务不同)；另一部分是目标区域的位置精修损失(需计算每一类下位置精修的便宜量)。具体的损失项展开有：

$$\begin{cases}L_{\text{clc}}(\boldsymbol{x}_n,y_n;\theta)=\dfrac{1}{S}\displaystyle\sum_{s=1}^{S}\text{loss}(\hat{\text{Lable}}_{x_n}^{(s)},\text{Lable}_{x_n-\text{target}}^{(s)})\\ L_{\text{loc}}(\boldsymbol{x}_n,y_n;\text{Refine}(\text{RP}_{x_n}))=\dfrac{1}{S}\displaystyle\sum_{s=1}^{S}\dfrac{1}{C}\sum_{c=1}^{C}R(\text{RP}_{x_n}^{(s)},P_{x_n-\text{target}}^{(s)})\end{cases} \tag{3.28}$$

式(3.28)中的分类损失函数可以用交互熵损失来展开，位置精修损失中的非线性函数仍用式(3.26)中的定义。

(4) 求解：

求解优化目标函数式(3.24)和式(3.27)，采用随机梯度下降的方法进行端到端的训练，两个优化目标函数交替进行优化，其中图 3.13 中的虚线框中为共享卷积层，减少计算量。由于篇幅所限，后面关于目标检测的应用会具体给出优化与参数设置以及实现细节等，这里不再赘述。

3.2.2　学习算法及训练策略

接下来，我们给出深度卷积神经网络的一些训练技巧，从数据的预处理、网络模型的参数初始化、训练过程中的学习速率及激活函数特性分析、正则化约束等角度总结一些实用的训练技巧。

1. 数据预处理

数据的预处理包括输入数据的预处理及隐层输出的归一化处理，其中输入数据的预处理常用的有：数据集去冗余，即给定训练数据集，计算其均值，然后数据集中的每一个数据减去均值，得到的新数据集作为网络的输入，该处理对于输入拓扑结构简单的数据集常常有效，例如基于深度学习平台 Caffe 下的手写体数据识别任务，若没有此操作，则模型的训练性能和测试性能(也称泛化性能)比较差。另外层级间的归一化处理，可以保持层级间信息传递的值域一致(而不是呈现逐层衰减)，起到加速运算的作用。需要注意的是，并不是所有的数据集在使用深度卷积神经网络时都进行数据的预处理(例如使用 GoogleNet 对 ImageNet 数据集进行分类的时候，就没有使用输入数据的预处理等)，除了这里提到的数据

预处理方法以外,图像的增强、修复、降噪等本质上可以提升数据的“质”,使得拓扑结构信息更为清晰、丰富。

2. 网络模型参数初始化

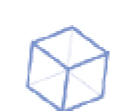
深度网络模型参数初始化的目的是减弱非凸优化目标函数对初值的依赖性,尽可能避免所求的解(即模型的参数)过早地陷入局部最优。深度卷积神经网络在没有引入逐层学习之前,关于模型参数的初始化主要有四种:零化(初始参数设置为零)、完全随机(服从于高斯分布)、带尺度约束的随机(尺度因子在−1 与 1 之间)和 Xaviar-glorot(不同分布下的半随机初始化)。利用手写体数据集 Mnist,网络模型为 VGG 网络,通过最后的损失函数和准确率的判断,带有尺度约束的 Xaviar-glorot 初始化参数的方式是最好的。除了以上四种常用的参数初始化外,还有利用各种变换所对应的(解析)函数的离散化来构造相应的滤波器集合,从该集合中随机选择滤波器来进行参数初始化的选取,例如 Gobor 变换所对应的 Gobor 函数(具有类初级视觉皮层的特性,即局部化、方向和多尺度特性等)、小波变换中的各种小波母函数、多尺度几何分析所对应的各种二代小波(例如曲波、脊波、楔形波和轮廓波等)。另外,对于深度卷积神经网络超参数的选择也非常重要,例如一般倾向于使用小滤波器(如 3×3 的尺寸),小步长(Stride)和补零(Zero_padding),这样就不会减少参数数量,从而提升整个网络的准确率,另外常用的池化尺寸是 2×2,可以在保持平移不变性的同时,有效地降低参数数量。

3. 训练阶段学习速率及激活函数特性分析

学习速率的调整可以通过验证集来实现,若发现验证集上的表现(包括损失函数和准确率)不再提升,将学习率除以 batch size,这样可以通过增大 batch size 而达到验证集上的损失函数降低、准确率提升的效果。常用的激活函数有修正线性单元 ReLU,相比于 Sigmoid、Tanh、Softplus 等激活函数,它具有不饱和、计算快、稀疏等特性。进一步,对修正线性单元的改进提出了 Leaky ReLU(给 ReLU 的负半轴加一个小斜坡)、Parametric ReLU、Randomized ReLU(负半轴上函数的斜率在训练时是随机的,测试时固定)等非线性函数,各有优势。

4. 正则化约束

众所周知,深度神经网络模型的训练依赖于数据量(自然获取、人工裁剪与旋转、生成式对抗网络获取),刻画网络的优劣则是泛化性能(外插或预测)。提升泛化性能常遇到的问题便是过拟合。防止过拟合有效的策略有 DropOut 正则化(它指随机地让网络某些隐层节点暂时不工作,不工作的那些节点可以暂时认为不是网络结构的一部分)。相比较其他正则化策略,DropOut 通过概率来刻画某一隐层上隐单元的激活特性(删减),且不同 Epoch 下的激活特性不同(动态变化),具有平衡数据量与模型参数量的作用,与稀疏性一样,合理地使用可以改良模型的性能(合理性:不是所有策略在每一层上都使用),与它相似的是 DropConnect(保留),关于 Dropout,通常在训练阶段有 Dropout,测试阶段不进行 Dropout,



可以看作是所有(指数级数目的)子图测试结果的平均,类似组合方法,Dropout 比例可以设置为 0.5,也可以在验证集上验证得出。除了 DropOut 正则化策略以外,还可以使用 L2/L1 正则化(约束权值连接矩阵)。

更多关于深度卷积神经网络的训练技巧可以参考印度深度学习专家 Rishabh Shukla 总结的 15 条训练建议,包括训练数据、选择合适的激活函数、隐层单元及层数的设计、权值初始化、学习率、格式超参数搜索、优化算法、逐层学习和批量尺寸、DropOut 正则化、迭代次数、自动微分求导等实践的经验总结以及理论分析。

备注:关于学习算法及训练策略的其他方法请参考:http://rishy.github.io/ml/2017/01/05/how-to-train-your-dnn.

3.2.3　模型的优缺点分析

深度卷积神经网络的核心模块是卷积流(即卷积、池化、非线性和批量归一化等操作),相比较于全连接形式的深度前馈神经网路,模型的优势在于稀疏(局部)连接、权值共享(网络连接结构)和特征图的平移不变性(刻画层级特征的统计特性)等特性;不论是 LeNet 网络、AlexNet 或者 VGG 网络,再或者是 Maxout 网络、ZF 网络、Overfeat 网络、Network in Network,还是 GoogleNet、深度残差网络、深度分形网络等,这些深度卷积神经网络的模型都具有通用的设计模式或者特点,例如结构上需符合应用,多通道(注重模型架构中的分支数量,而不是继续增加深度),简洁性(使用更少类型的层以保持网络尽可能简单,如深度分形网络),塔式结构(整体的平滑的下采样),对称性,以及技巧上的批量归一化输入,过拟合(引入正则项,包括有噪声数据的使用提升网络的泛化性能)等。

备注:支持深度学习平台 Caffe 的模型可视化网页版 Netscope 链接地址为 http://ethereon.github.io/netscope/quickstart.html.

该网页可以可视化上面所提到的各种网络及给出网络模型中每一模块的参数设置(深灰色框),同时也可以可视化设计的网络并检查网络的设计和前向计算是否正确。与 Caffe 自带的 draw_net.py 绘制网络模型结构相比,Netscope 绘制的网络模型简洁、直观,图 3.14 中由于 VGG16 和 ResNet50 网络模型结构尺寸较大,仅给出部分结构。

下面不再对上面所提到的网络逐一进行详细的描述与分析,仅介绍几种常用且赢得 ImageNet 物体识别挑战的卷积神经网络结构,并简要地给出成功应用的领域介绍与性能指标、网络模型的特点。

备注 1:ImageNet 是一个拥有超过 1500 万张带标签的高分辨率图像的数据集,这些图像分属于大概 22 000 个类别。这些图像是从网上收集,并使用 Amazon Mechanical Turk 群众外包工具来人工贴标签的。ILSVRC 使用 ImageNet 的一个子集,分为 1000 种类别,每种类别中都有大约 1000 张图像。总之,大约有 120 万张训练图像,50 000 张验证图像和 150 000 张测试图像。

备注 2:其中图 3.15 给出的 Top5 错误率是指对于 ImageNet 图像通常有 1000 个可能的类别,对每幅图像可以猜 5 次结果(预测 5 个类别标签),若其中有任何一次预测对了,结果都算正确,当 5 次全都错了的时候,才算预测错误,这时候的分类错误率就叫 top5 错误率。

图 3.14　NetScope 绘制 AlexNet 网络、VGG16 网络、深度残差网络 ResNet50

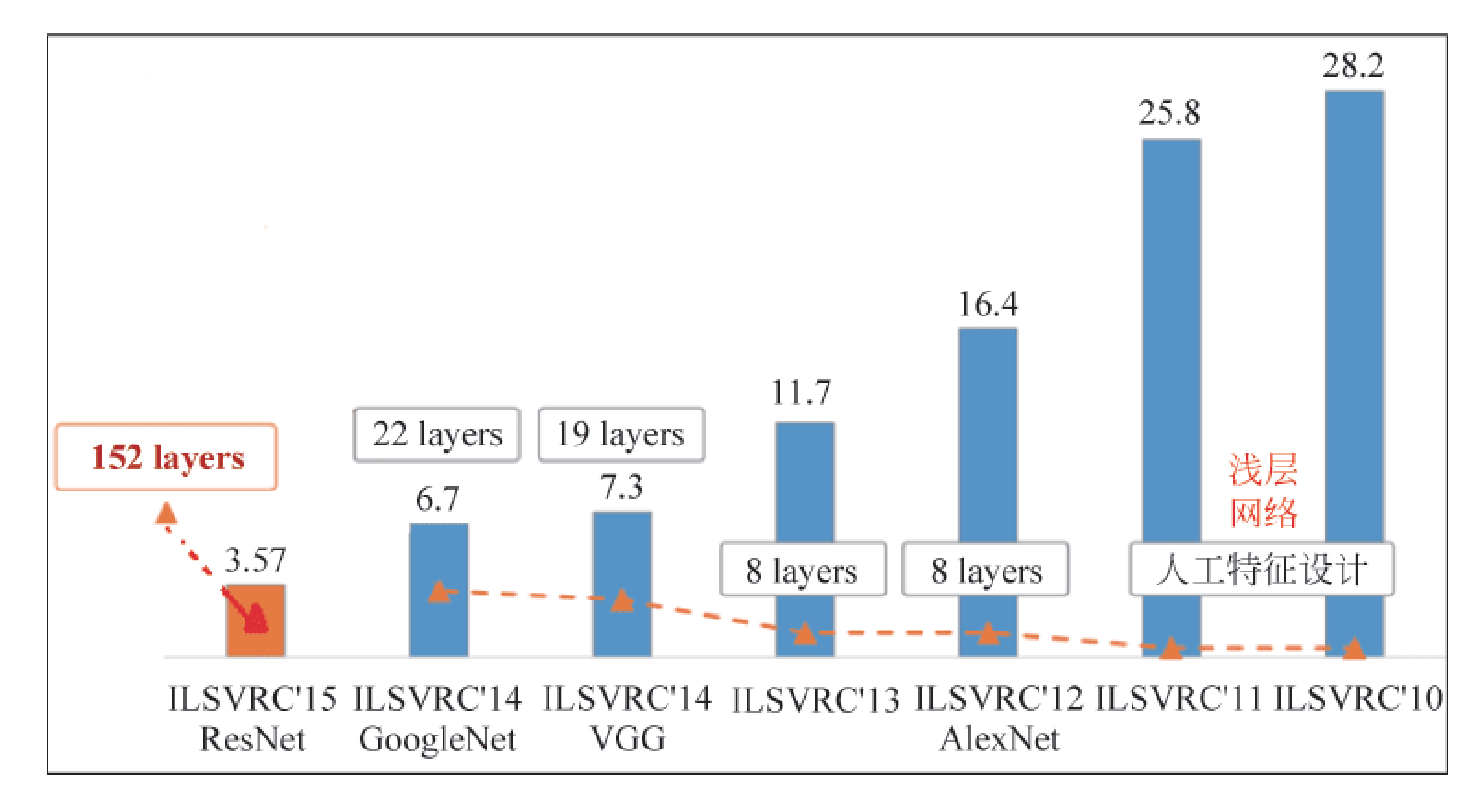
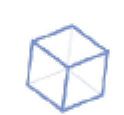


图 3.15　ILSVRC 历年的 Top5 错误率



1. AlexNet 网络

AlexNet 网络模型在 ILSVRC 2012 图像分类任务上赢得冠军，该模型的优势在于：引入多种技巧与策略(如 Dropout、数据扩张、局部响应归一化和重叠池化、ReLU 激活函数)解决过拟合，并且可以利用多 GPU 加速计算。

2. VGG、GoogleNet 网络

VGG 和 GoogleNet 是 ILSVRC 2014 竞赛的双雄，这两类模型结构有一个共同特点：层级开始走向"极深"，跟 GoogleNet 不同的是，VGG 继承了 LeNet 以及 AlexNet 的一些框架，尤其是跟 AlexNet 框架非常像。另外，注意使用更多的卷积、更多的层次可以得到更好的结构，但是随着卷积层的逐渐加深，准确率的提升也愈加困难。

3. 深度残差网络

深度残差网络的特点：一是网络层级较深、但每一隐层较瘦，可以控制参数的数量；二是存在层级，特征图个数逐层递进，保证输出特征表达能力；三是使用了较少的池化层，大量使用下采样，提高传播效率；四是没有使用 Dropout，利用批量归一化和全局平均池化进行正则化，加快了训练速度；五是层数较高时减少了 3×3 卷积个数，并用 1×1 卷积控制了 3×3 卷积的输入输出特征图数量，称这种结构为"瓶颈"。六是深度网络受梯度弥散问题的困扰，批量归一化、ReLU 等手段对梯度弥散缓解能力有限，而深度残差网络中的单位映射的残差结构可以从本源上杜绝该问题。值得指出的是，深度卷积神经网络在深度学习的历史中发挥了重要的作用，将脑科学最新研究获得的深刻机理成功用于模式识别领域并大获成功，成为影响深刻的深度学习模型之一。

3.3　深度反卷积神经网络

深度反卷积神经网络是一种基于卷积稀疏编码模块的深度学习模型，众所周知，常见的深度网络模型是由单层的网络叠加而成的，而单层网络按照编码类型可以分为三类，一是具有编码(或分析)与解码(或合成)的过程，如各种自编码网络、受限玻尔兹曼机等；二是只具有合成的过程，如稀疏编码、稀疏表示、卷积稀疏编码等；三是只具有分析的过程，如普通的前馈神经网络等；下面用三个小节来分析深度反卷积神经网络。

3.3.1　卷积稀疏编码

与稀疏编码或稀疏表示不同的是：卷积稀疏编码首次将卷积操作与稀疏编码结合起来，充分发挥卷积操作的计算优势，同时从解码(合成)的角度实现隐层特征的求解。首先解释反卷积的概念，若已知 $\boldsymbol{x}\in\mathbb{R}^n$，$\boldsymbol{h}\in\mathbb{R}^m$ 且关系如下：

$$h = w * x \in \mathbb{R}^m \tag{3.29}$$

求解 $w \in \mathbb{R}^s$ 的过程便为反卷积运算，其中 $m=s+n-1$，注意这里描述的卷积为式(3.1)中的Full 卷积，其他的卷积操作类似。其次，通过图 3.16 所示，给出卷积稀疏编码的公式描述：

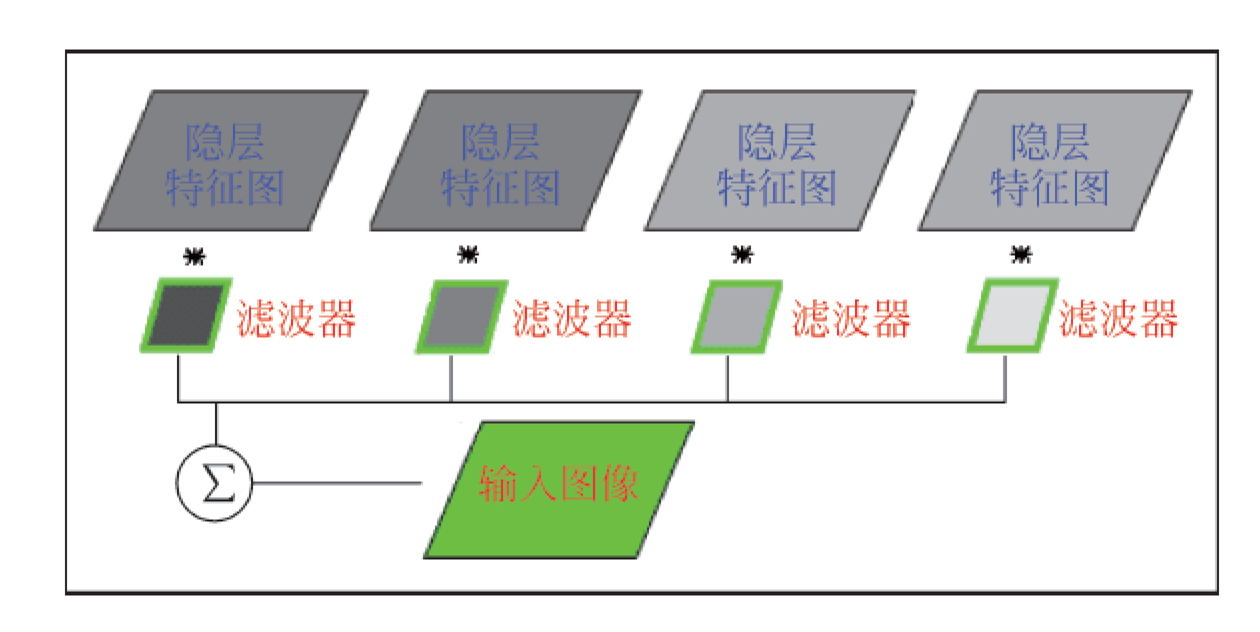


图 3.16　卷积稀疏编码图示

假设，输入图像 $x \in \mathbb{R}^{n\times m}$，超参数设定为：期望输出 4 个特征图，记为 $h^{(i)} \in \mathbb{R}^{u\times v}$；另外 4 个滤波器尺寸也一致，记为 $w^{(i)} \in \mathbb{R}^{s\times r}$，其中 $i=1,2,3,4$。进一步输入与输出特征图之间的关系为(注意输入与输出特征图之间为“全连接”，即每一个隐层特征图都与输入相连，如前面 LeNet5 网络模型中的特征图连接关系)：

$$x = \sum_{i=1}^{4} w^{(i)} * h^{(i)} \tag{3.30}$$

期望根据大量的输入图像数据集，能够学出滤波器与隐层特征图(与稀疏表示中的 KSVD 算法一样，严格意义上，此处合成情形下的滤波器应该为“字典”)。最后从数据、模型、优化目标函数和求解四个方面陈述卷积稀疏编码。

1. 数据

$$\{x^{(t)} \in \mathbb{R}^{n\times m}\}_{t=1}^{N} \tag{3.31}$$

由于卷积稀疏编码是无监督的学习方式，所以只需要输入数据集便可。

2. 模型

输入与输出(即隐层特征图)之间的关系为式(3.30)，这里不再赘述。

3. 优化目标函数

依据数据和模型，得到如下的优化目标函数(包括损失项和正则项)：

$$\min J(\{w^{(i)}, h^{(i)}\}_{i=1}^{4}) = \frac{1}{N}\sum_{t=1}^{N}\left\| x^{(t)} - \sum_{i=1}^{4} w^{(i)} * h_t^{(i)} \right\|_F^2 + \lambda \cdot \sum_{i=1}^{4} \| w^{(i)} \|_F^2 \tag{3.32}$$

其中 $h_t^{(i)}$ 为第 t 幅输入 $x^{(t)}$ 所得到的第 i 幅隐层特征图，λ 为拉格朗日因子。注意除了可以加入基于能量的正则项避免过拟合现象外，还可以加入隐层特征图的稀疏正则约束，这里不再赘述。

4. 求解

求解算法根据 ADMM 或参考 K-SVD 算法，即固定滤波器时，更新隐层特征图，再固定隐层特征图时，更新滤波器，交替进行直至收敛。

3.3.2　深度反卷积神经网络

基于卷积稀疏编码来构建深度网络模型的层级结构，称为深度反卷积神经网络，下面先给出深度反卷积神经网络中层级结构之间的关系结构图（图 3.17），并利用公式来具体描述这种关系：

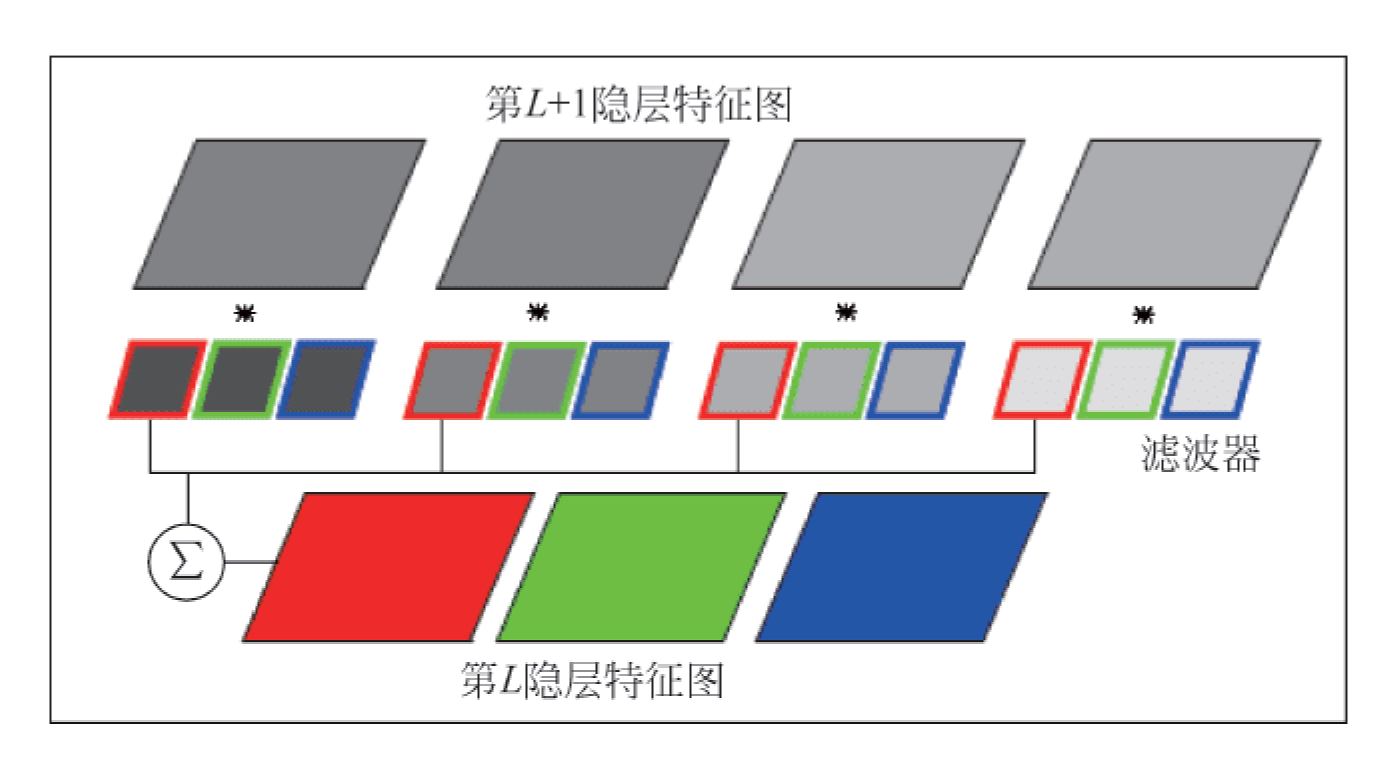


图 3.17　深度反卷积神经网络中的相邻层级连接结构

注意第 L 层与第 $L+1$ 层之间的连接仍为“全连接”，如第 L 层的红色特征图与第 $L+1$ 层都有连接关系，所以全连接带来的滤波器的个数为 12 个。对应有如下的层级（前向）传递关系：

$$\boldsymbol{h}_L^{(i)} = \sum_{j=1}^{4} \boldsymbol{w}_{L,L+1}^{(i,j)} * \boldsymbol{h}_{L+1}^{(j)} \tag{3.33}$$

其中 $i=1,2,3$，符号 $\boldsymbol{w}_{L,L+1}^{(i,j)}$ 为第 L 层第 i 个特征图与第 $L+1$ 层第 j 个特征图之间的滤波器（严格意义上称为字典）。

深度反卷积神经网络仍沿用深度卷积神经网络的架构，即卷积稀疏编码（替代卷积的功能，产生隐层特征图）、池化、非线性和批量归一化。学习隐层的特征图可以通过合成方式下的“逐层学习”方式求得，注意与自编码方式下逐层学习初始化参数（层级连接权值）有所不同。假设每一幅输入图像 $\boldsymbol{x}^{(t)}$ 前向计算至第 L 层的特征图记为 $\boldsymbol{h}_L^{(i)}(\boldsymbol{x}^{(t)})$，通过如下的优化公式可以求解第 $L+1$ 层上的特征图：

$$\min \frac{1}{N}\sum_{t=1}^{N}\sum_{i=1}^{3}\left\|\boldsymbol{h}_L^{(i)}(\boldsymbol{x}^{(t)}) - \sum_{j=1}^{4}\boldsymbol{w}_{L,L+1}^{(i,j)} * h_{L+1}^{(j)}(x^{(t)})\right\|_F^2 + \lambda \cdot \sum_{i=1}^{3}\sum_{j=1}^{4}\|\boldsymbol{w}_{L,L+1}^{(i,j)}\|_F^2 \tag{3.34}$$

后面再对第 $L+1$ 层的特征图进行池化、非线性（ReLU）和批量归一化、全连接层等操作，与深度卷积神经网络一致，这里不再赘述。这里给出基于 CaffeNet（与 AlexNet 结构相同，需将 AlexNet 网络中所有池化操作与非线性 ReLU 操作次序颠倒便可以得到 CaffeNet 网络模型）的深度反卷积神经网络模型（见图 3.18）。

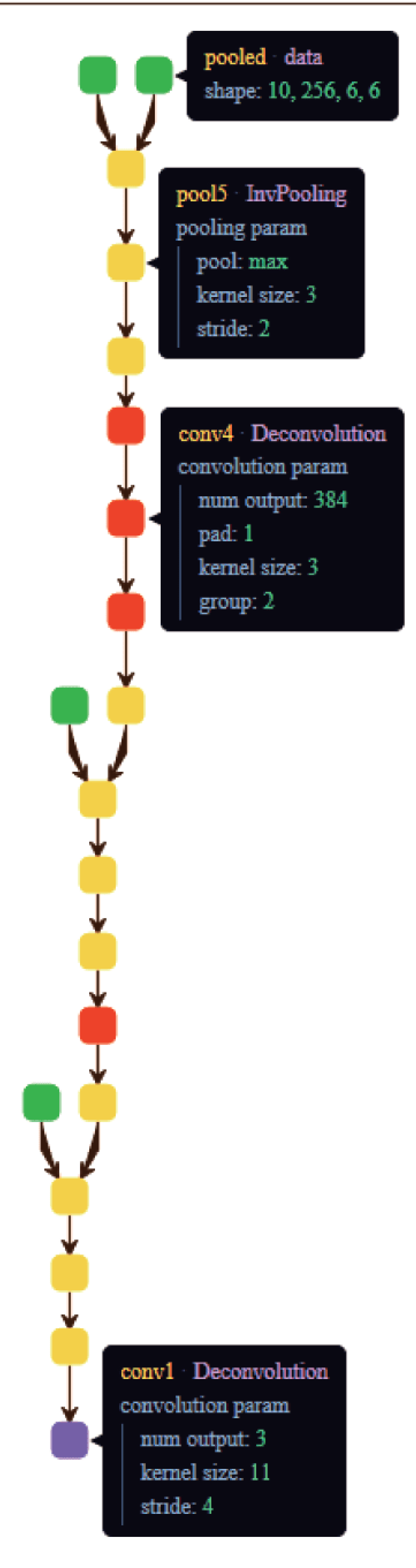


图 3.18　基于 CaffeNet 的深度反卷积神经网络模型

备注：关于深度反卷积神经网络更为详细的解释与 Caffe 代码可参考网页 https://github.com/piergiaj/caffe-deconvnet.

3.3.3　网络模型的性能分析与应用举例

深度反卷积神经网络与深度卷积神经网络的应用场景一样，如分类任务、目标检测和语义分割等，当然它还可以应用至图像复原任务；但由于卷积稀疏编码受限于训练和测试阶段，导致前向传播计算的速度较慢。另外，可以利用反卷积操作（卷积稀疏编码）可视化理解深度卷积神经网络；本质上，由于卷积操作仍为线性操作，所以卷积稀疏编码与稀疏编码从形式上是一致的，进而深度反卷积神经网络与稀疏层次目标识别网络 S-HMAX（第 1 章中的内容）是一致的。

3.4　全卷积神经网络

全卷积神经网络（Fully Convolutional Networks，FCN）对输入图像进行像素级分类，解决了语义级别的图像分割问题，与经典的卷积神经网络在若干卷积流后使用全连接层得到固定长度的特征向量进行分类（本质上，是一种图像级别的语义理解，分类器设计常用 Softmax 函数）不同，全卷积神经网络可以接受任意尺寸的输入图像，引入反卷积操作对最后一个卷积层上的特征图进行上采样（需将卷积神经网络中的全连接层也改成卷积层，顾名思义网络结构中没有全连接层，都为卷积流架构），使特征图恢复到与输入图像相同的尺寸，从而可以对每一个像素产生一个预测，同时保留原始输入图像中的空间信息，最后在上采样的特征图上进行逐像素分类。下面我们用两小节细述全卷积神经网络。

3.4.1　网络模型的数学刻画

首先通过一个简单的例子来理解全卷积神经网络的概念，该例子使用的网络结构如图 3.19 所示。

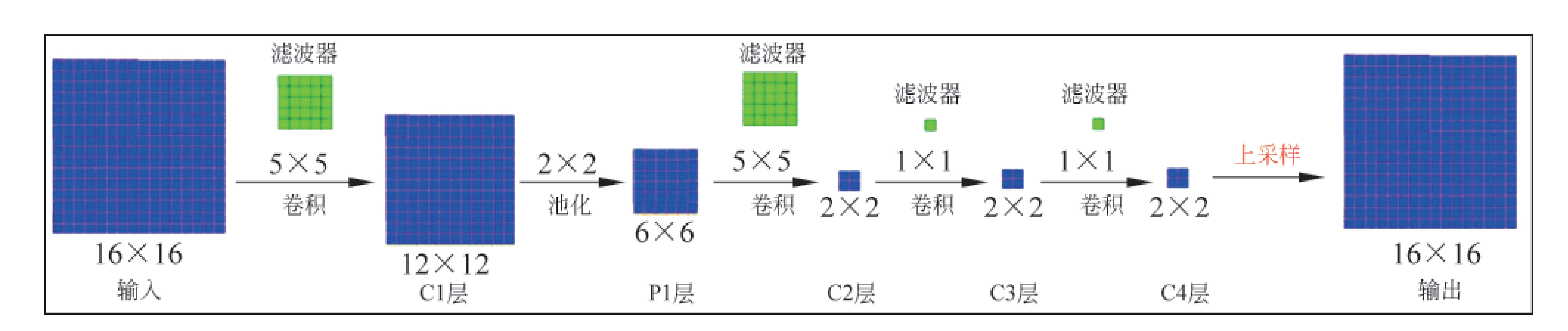


图 3.19　全卷积神经网络流图

可以看出，从输入向输出传输的过程中，图 3.19 中没有类似 LeNet、AlexNet 网络中的全连接层操作，其中核心操作为上采样，即如何依据 C4 实现输出的尺寸与输入一致？最为

简单的上采样策略是双线性插值，但有可能损失很多的细节信息，使得输出逐像素刻画输入的特性较为粗糙。假设该网络训练好以后，所有层级上的滤波器固定，同时上采样的参数也已固定，不同尺寸的输入将会导致 C4 的尺寸不一，那么仅靠固定的上采样操作是不可能完成输出与输入尺寸一致的要求，所以通常需要在上采样操作后加入裁剪层(Crop Layer)，这样全卷积神经网络便可以实现对输入图像尺寸不限制的要求。

需要注意以下四点，一是全卷积神经网络要求输入与输出的尺寸一致，但并没有要求二者的拓扑结构信息一致(即输入与输出的内容尽可能的一致)，也许输入为 RGB 彩色图 $\boldsymbol{x}\in\mathbb{R}^{16\times16\times3}$，但输出却为 $\boldsymbol{y}\in\mathbb{R}^{16\times16\times10}$，即对于输入中的每一个像素都有输出与其对应：

$$\begin{cases}\text{FCN}:\boldsymbol{x}(u,v,1,:)\in\mathbb{R}^3\longrightarrow y(u,v,:)\in\mathbb{R}^{10}\\ u,v=1,2,\cdots,16\end{cases}\tag{3.35}$$

其中 $\boldsymbol{y}\in\mathbb{R}^{16\times16\times10}$ 中的 10 可以理解为 10 类，即像素级理解或分类问题；二是普通卷积神经网络中全连接层转化为卷积层的过程；三是上采样层可量化为反卷积的过程；四是影响输入图像随着层级的变化而缩小的操作包括卷积中的步长设置和池化操作中的半径与步长设置。

通过对全卷积神经网络的概念性初步理解，接下来根据应用实例，通过数学刻画整个流程，核心在于上采样操作与裁剪层的刻画，该实例使用的网络模型见图 3.20。

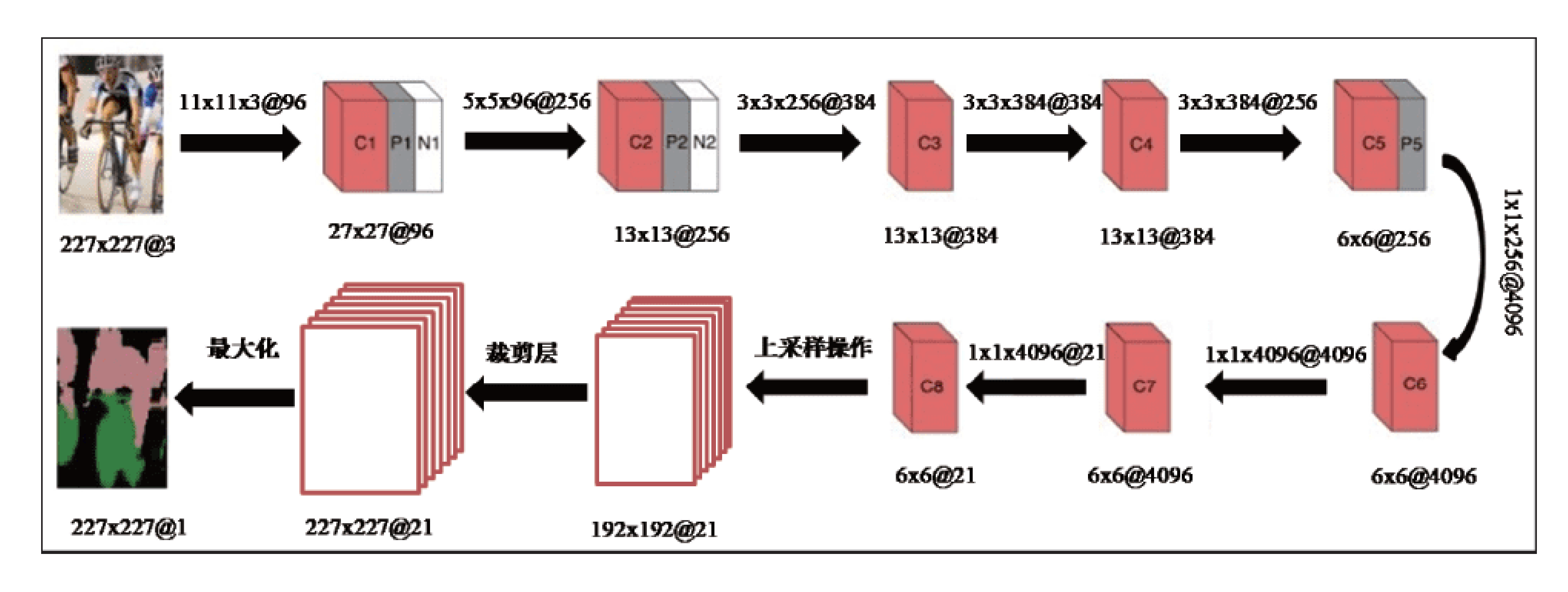


图 3.20　基于 AlexNet 网络结构(其中全连接改为卷积层)的像素级语义分割

注意该网络结构使得图像缩小的操作主要发生在：第一个卷积流后(卷积操作中的步长为 4，即 stride=4；池化半径为 2)图像缩小到原来的 $\frac{1}{8}$；第二个卷积流后(卷积操作不改变大小，即 stride=1；池化半径为 2)再缩小到原来的 $\frac{1}{2}$；第五个卷积流后(池化半径为 2)再缩小到原来的 $\frac{1}{2}$；共计缩小到原来的 $\frac{1}{32}$。所以上采样操作是在 C8 的基础上通过双线性插值运算得到特征图的大小为 192×192@21，注意特征图的个数 21 是类别数，即输入图像中的每个像素可划分为 21 类；由于尺寸与输入尺寸不一致所以需要裁剪层将其进一步扩

展至原图大小，最后对特征图中每一个位置根据通道实现（共计 21 维向量）最大化，即最大化数值所对应的指标便是该像素的类标。下面我们通过数据、模型、优化目标函数和求解四个方面来刻画基于 Alexnet 网络的像素级语义分割任务（其中 AlexNet 网络在全连接层之前的参数可以参考 Caffe 平台上的网络传输协议，这里不再赘述）。

1. 数据

$$\{\boldsymbol{x}^{(n)} \in \mathbb{R}^{227\times227\times3}, \boldsymbol{y}^{(n)} \in \mathbb{R}^{227\times227\times21}\}_{n=1}^{N} \tag{3.36}$$

其中输出 $\boldsymbol{y}^{(n)}$ 为输入所对应的语义分割后的结果，其尺寸大小与输入相同，但通道数不一样，例如第一个像素属于第二类，那么在这 21 个通道中，仅有第 2 个为 1，其他全为 0。

2. 模型

$$\begin{cases} \boldsymbol{X}_1 = C_8(\boldsymbol{x}, \theta_1) \in \mathbb{R}^{6\times6@21} \\ \boldsymbol{X}_2 = \mathrm{Upsampling}(\boldsymbol{X}_1, \theta_2) \in \mathbb{R}^{192\times192@21} \\ \boldsymbol{X}_3 = \mathrm{Crop}(\boldsymbol{X}_2, \mathrm{size}(\boldsymbol{x})) \in \mathbb{R}^{227\times227@21} \\ y(u,v) = \arg\max\limits_{1\leqslant s\leqslant 21} \boldsymbol{X}_3(u,v,s) \\ u,v = 1,2,\cdots,227 \end{cases} \tag{3.37}$$

注意最后输出的 $y(u,v)$ 记录下了应该将该像素分到哪一类，并根据该类的着色最后输出分割图像，依据像素的类别归属将其扩展至 227×227×21 的矩阵，作为对输入的像素级语义分割的预测结果。

3. 优化目标函数

$$\min_{\theta} J(\theta) = \frac{1}{N}\sum_{n=1}^{N} L(\boldsymbol{y}^{(n)}, \hat{\boldsymbol{y}}^{(n)}) + \lambda \cdot R(\theta) \tag{3.38}$$

其中损失项中可以利用基于能量的损失，也可以利用负对数似然（交叉熵）来实现，参数 θ 包括两部分，第一部分 θ_1 为各个卷积流中和分类器 Softmax 函数中的参数；第二部分为上采样中的参数 θ_2（若为固定模式，如输入在上采样之前都是缩小到原来的 1/32，那么参数便可以不用学习），另外 $R(\theta)$ 为正则项，防止过拟合现象。

4. 求解

求解算法利用随机梯度下降，与卷积神经网络中的优化求解一致，当然模型的稳定性与收敛性取决于数据量和好的初始参数的选取，之前在卷积神经网络中的各种策略与方法仍可以使用。

3.4.2　网络模型的性能分析及应用举例

模型的优点有：一是训练一个端到端的全卷积神经网络模型，利用卷积神经网络很强

的学习能力，能够得到较准确的结果，与以前的基于卷积神经网络的方法相比不用再对输入或者输出做处理；二是直接使用现有的卷积神经网络模型，如 AlexNet、VGG16、GoogleNet，只需将其中的全连接层改为卷积层并采用上采样和裁剪操作，即可实现网络的架构；三是不限制输入图片的尺寸，不要求图片集中所有图片都是同样尺寸。模型的缺点：和期望输出相比，该方法容易丢失较小的目标。对于模型的改进便引入了多尺度精细化策略，即不用一次性地将特征图 C_8 上采样至 32 倍，可以先上采样 2 倍，结合 C_4 的信息，再上采样 16 倍；实验发现后者分批分次进行上采样并利用之前特征图上的信息，得到的结果能够保留较小目标的细节。

目前，全卷积神经网络主要应用的场景为目标检测中的定位（即候选框的提取，类似于“注意机制”下的显著性检测任务）和语义分割任务等。

参考文献

[3.1] Krizhevsky A，Sutskever I，Hinton G E. ImageNet classification with deep convolutional neural networks[C]. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012：1097-1105.

[3.2] Bouvrie J. Notes on Convolutional Neural Networks[J]. Neural Nets，2006.

[3.3] Simonyan K，Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science，2014.

[3.4] Lecun Y，Bottou L，Bengio Y，et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE，1998，86(11)：2278-2324.

[3.5] Simard P Y，Steinkraus D，Platt J C. Best practices for convolutional neural networks applied to visual document analysis[C]. International Conference on Document Analysis and Recognition. IEEE Computer Society，2003：958.

[3.6] Lawrence S，Giles C L，Tsoi A C，et al. Face recognition：a convolutional neural-network approach[J]. IEEE Transactions on Neural Networks，1997，8(1)：98-113.

[3.7] Maggiori E，Tarabalka Y，Charpiat G，et al. Convolutional Neural Networks for Large-Scale Remote Sensing Image Classification[J]. 2016.

[3.8] Sermanet P，Eigen D，Zhang X，et al. OverFeat：Integrated Recognition，Localization and Detection using Convolutional Networks[J]. ePrint arXiv，2013.

[3.9] Szegedy C，Liu W，Jia Y，et al. Going deeper with convolutions[J]. 2015：1-9.

[3.10] Chen L C，Papandreou G，Kokkinos I，et al. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs[J]. Computer Science，2014(4)：357-361.

[3.11] Girshick R，Donahue J，Darrell T，et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation[C]. Computer Vision and Pattern Recognition. IEEE，2013：580-587.

[3.12] Karpathy A，Toderici G，Shetty S，et al. Large-Scale Video Classification with Convolutional Neural Networks[C]. IEEE Conference on Computer Vision and Pattern Recognition. IEEE

Computer Society，2014：1725-1732.

[3.13] Kim Y. Convolutional Neural Networks for Sentence Classification[J]. Eprint Arxiv，2014.

[3.14] Sainath T N，Kingsbury B，Saon G，et al. Deep Convolutional Neural Networks for large-scale speech tasks[J]. Neural Networks the Official Journal of the International Neural Network Society，2015，64：39-48.

[3.15] Wang T，Wu D J，Coates A，et al. End-to-end text recognition with convolutional neural networks[C]. International Conference on Pattern Recognition. IEEE，2012：3304-3308.

[3.16] Ji S，Yang M，Yu K. 3D convolutional neural networks for human action recognition[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence，2013，35(1)：221-231.

[3.17] 李彦冬，郝宗波，雷航. 卷积神经网络研究综述[J]. 计算机应用，2016，36(9)：2508-2515.

[3.18] Zeiler M D，Fergus R. Visualizing and Understanding Convolutional Networks[J]. 2013，8689：818-833.

[3.19] Razavian A S，Azizpour H，Sullivan J，et al. CNN Features Off-the-Shelf：An Astounding Baseline for Recognition[J]. 2014：512-519.

[3.20] Kalchbrenner N，Grefenstette E，Blunsom P. A Convolutional Neural Network for Modelling Sentences[J]. Eprint Arxiv，2014，1.

[3.21] Abdel-Hamid O，Mohamed A R，Jiang H，et al. Convolutional Neural Networks for Speech Recognition[J]. IEEE/ACM Transactions on Audio Speech & Language Processing，2014，22(10)：1533-1545.

[3.22] Silver D，Huang A，Maddison C J，et al. Mastering the game of Go with deep neural networks and tree search[J]. Nature，2016，529(7587)：484.

[3.23] Zeiler M D，Fergus R. Stochastic Pooling for Regularization of Deep Convolutional Neural Networks[J]. Eprint Arxiv，2013.

[3.24] Chatfield K，Simonyan K，Vedaldi A，et al. Return of the Devil in the Details：Delving Deep into Convolutional Nets[J]. Computer Science，2014.

[3.25] He K，Sun J. Convolutional neural networks at constrained time cost[C]. Computer Vision and Pattern Recognition. IEEE，2014：5353-5360.

[3.26] Springenberg J T，Dosovitskiy A，Brox T，et al. Striving for Simplicity：The All Convolutional Net[J]. Eprint Arxiv，2014.

[3.27] Long J，Shelhamer E，Darrell T. Fully convolutional networks for semantic segmentation[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence，2017，39(4)：640-651.

[3.28] Bengio Y，Lecun Y. Convolutional Networks for Images，Speech，and Time-Series[J]. 1995.

[3.29] He K，Zhang X，Ren S，et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence，2014，37(9)：1904-1916.

[3.30] Donahue J，Hendricks L A，Guadarrama S，et al. Long-term recurrent convolutional networks for visual recognition and description[C]. Computer Vision and Pattern Recognition. IEEE，2015：2625-2634.

[3.31] Lecun Y，Kavukcuoglu K，Farabet C. Convolutional Networks and Applications in Vision[J]. 2010，14(5)：253-256.

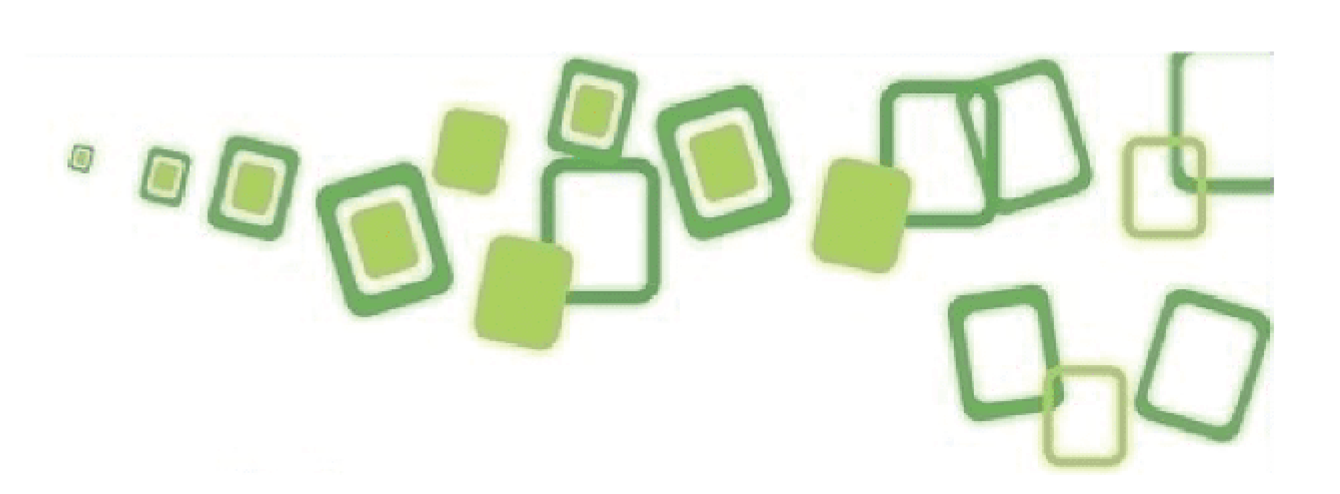
[3.32] Simonyan K, Vedaldi A, Zisserman A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps[J]. Computer Science, 2013.

[3.33] Jain V, Seung H S. Natural Image Denoising with Convolutional Networks[J]. Advances in Neural Information Processing Systems, 2008: 769-776.

[3.34] Oquab M, Bottou L, Laptev I, et al. Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks[C]. Computer Vision and Pattern Recognition. IEEE, 2014: 1717-1724.

[3.35] Sainath T N, Mohamed A R, Kingsbury B, et al. Deep convolutional neural networks for LVCSR [C]. IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013: 8614-8618.

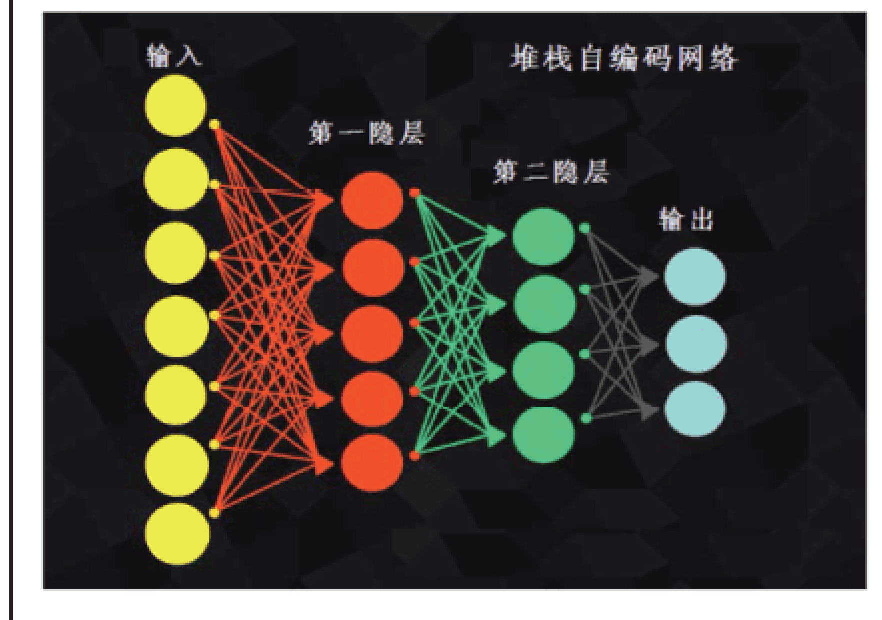
第4章

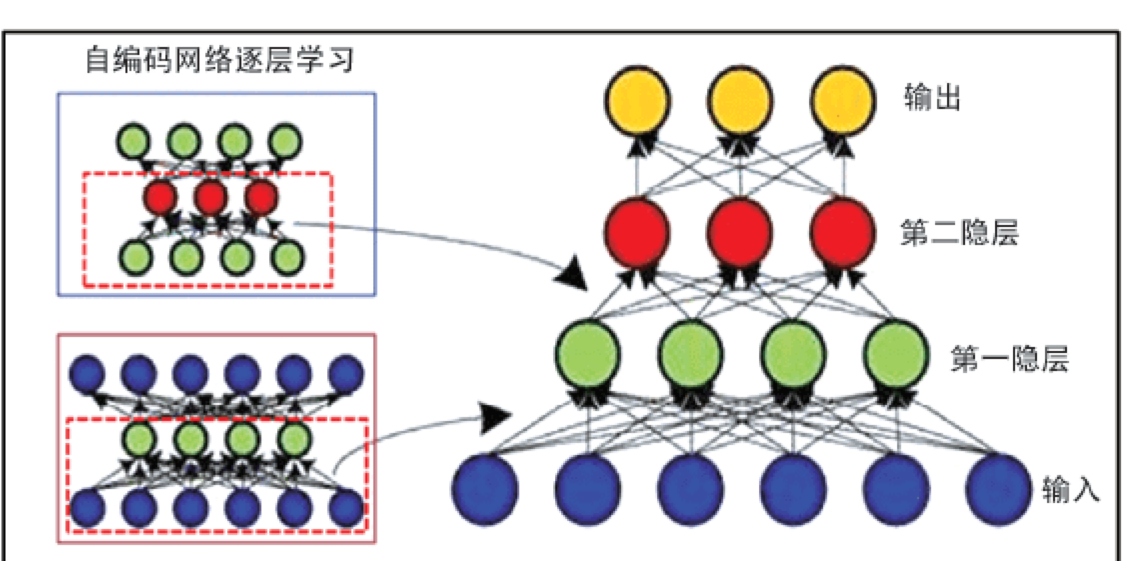


深度堆栈自编码网络

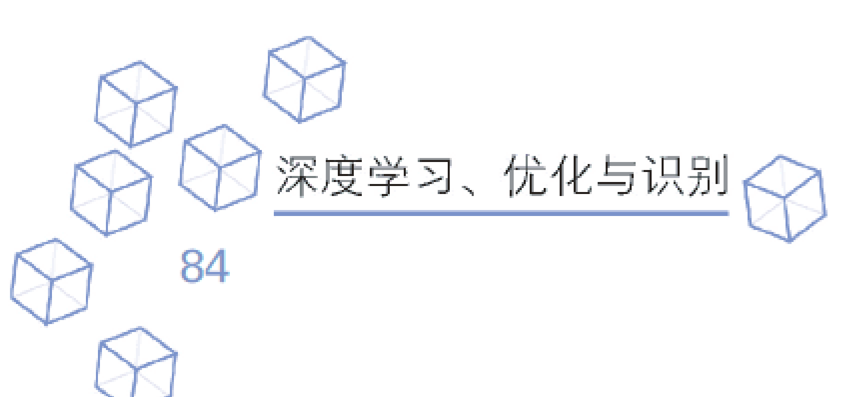
CHAPTER 4

深度堆栈网络——自编码网络、受限波尔兹曼机





$x \xrightarrow{f} X \xrightarrow{g} \hat{x}$ 逐层学习+精调



4.1 自编码网络

深度神经网络随着层级的加深，导致优化目标函数为非凸优化问题，依赖于初值的选取，选择较好时，可以避免过早地陷入局部最优，求得稳定状态下的参数逼近最优参数；若选择不好时，网络模型易出现欠拟合现象(即训练误差下降慢导致网络性能差)等；如何能够避免(优化目标函数所对应的)可行域上大量的鞍点与局部极值点？为解决这一问题，Hinton 等人提出使用无监督预训练方法优化网络权值的初值，再使用少量有类标的数据对权值进行微调，拉开了深度学习的序幕。另外，脑科学的研究发现，人脑具有一个深度结构，并且对外界事物的认知过程是逐层进行、逐步抽象的。例如，人的视觉系统对信息的处理是分级的，并显示出一系列的生物学功能区域，在每一个这种区域中包含一个输入的表示和从一个到另一个的信号流，根据信号流的方向，不同功能区对应着的层级表征不断被抽象，同时层级之间的响应特性不断被强化。下面分两小节详述 Hinton 等人的方法，同时引出深度堆栈网络的核心——自编码网络。

4.1.1 逐层学习策略

逐层学习策略，顾名思义，对深度神经网络层级间的参数进行“剖分”式学习，即将相邻层级视为浅层神经网络，可充分发挥浅层神经网络的学习优势(凸优化)，进一步，获取初始化参数后的层级通过“复合”(堆栈)形成深度神经网络，可以大大地节省计算存储资源和时间，提高网络模型的泛化性能。通常，基于逐层学习策略的参数初始化方法包括以下三种形式：一是分析形式(如独立成分分析、主成分分析等)；二是合成形式(如稀疏编码/稀疏表示，卷积稀疏编码等)；三是分析合成形式(如基于三层前馈神经网络的各种自编码网络，受限玻尔兹曼机和玻尔兹曼机等)。值得指出的是：深度神经网络参数初始化的方式不局限于上面给出的，还有非学习方式下的选取，例如基于 Gobor 变换、小波变换和多尺度几何分析等构造滤波器组的集合，随机从该滤波器组集合中选取若干滤波器，赋值给层级间的权值矩阵；以及服从某种分布下的参数半随机化赋值等。图 4.1 为基于逐层学习策略的深度神经网络训练模式。

4.1.2 自编码网络

自编码网络是指保持输入与输出尽可能一致(通过信息损失来判定)的情形下，实现无监督方式下的隐层特征提取与参数学习；其核心在于训练方式——无监督学习，以及实现方式——浅层神经网络(凸优化理论)，以及刻画方式——隐层特征的维数(通常，升维对应稀疏性，降维对应压缩)；对于深度神经网络而言，最终目的在于参数学习。本小节将详述基于三层(即包括输入层、隐层和输出层)前馈神经网络的自编码实现及理解：首先给出其网络结构(见图 4.2)。

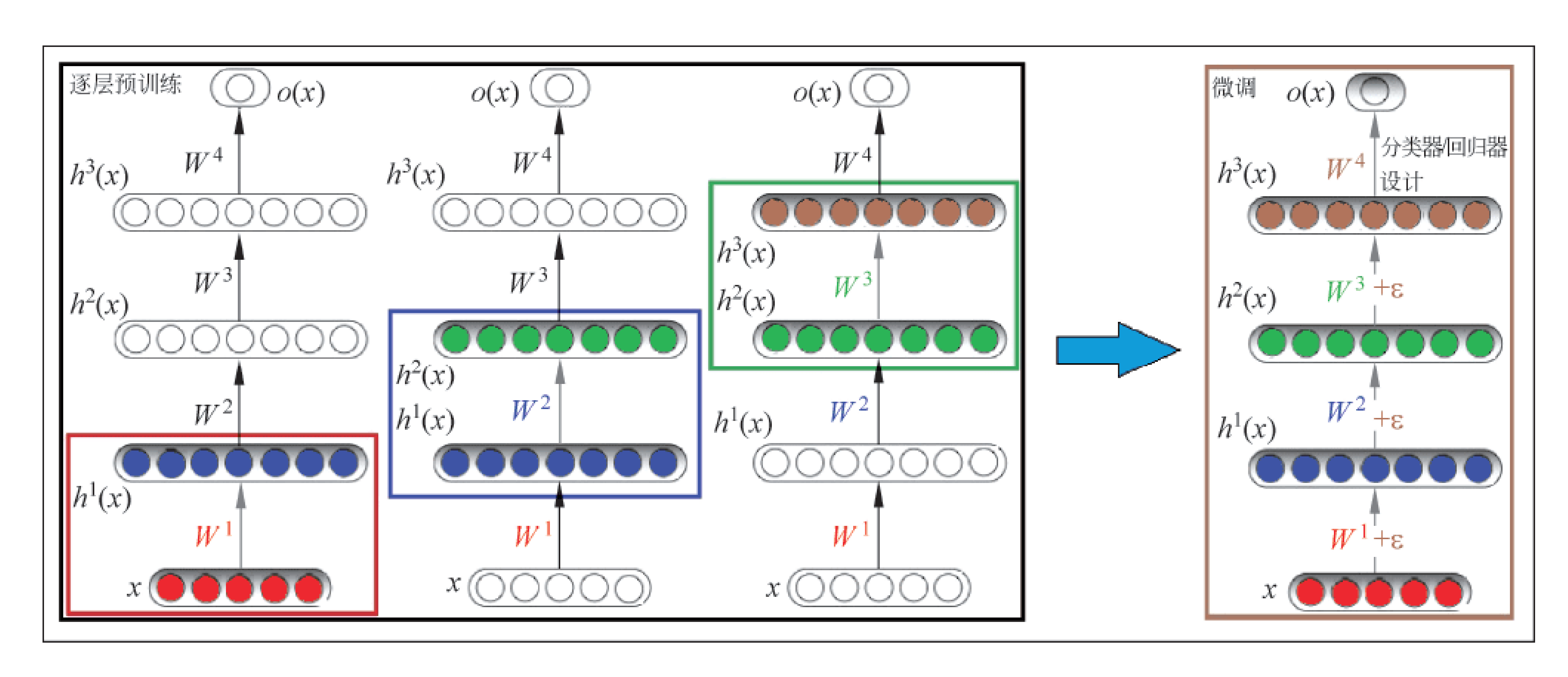


图 4.1　基于逐层学习策略的深度神经网络训练模式

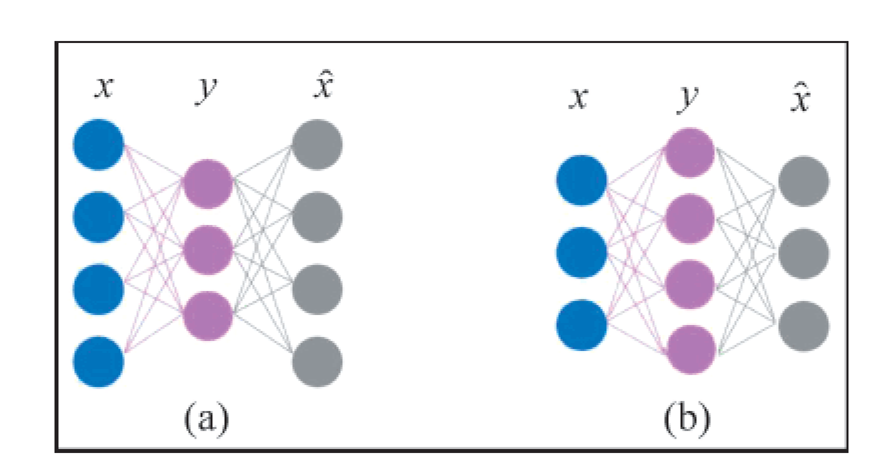


图 4.2　基于三层前馈神经网络的自编码结构

注意图 4.2 中，不论是中间隐层的特征维数上升或下降，对应的网络层级结构都是前向传播、拓扑无环结构。

1. 数据

$$\{\boldsymbol{x}^{(n)} \in \mathbb{R}^u\}_{n=1}^N \tag{4.1}$$

输入数据便是期望的输出。

2. 模型

$$\begin{cases} \boldsymbol{X} = \sigma_a(\boldsymbol{W}_a \cdot x + \boldsymbol{b}_a) \in \mathbb{R}^v \\ \hat{\boldsymbol{x}} = \sigma_s(\boldsymbol{W}_s \cdot \boldsymbol{X} + \boldsymbol{b}_s) \in \mathbb{R}^u \end{cases} \tag{4.2}$$

其中，分析（编码）阶段的参数为（$\boldsymbol{W}_a \in \mathbb{R}^{v\times u}$，$\boldsymbol{b}_a \in \mathbb{R}^v$），另外激活（非线性）函数为 $\sigma_a(\cdot)$，这里字母“a”为分析 analysis 的首字母；合成（解码）阶段的参数为（$\boldsymbol{W}_s \in \mathbb{R}^{u\times v}$，$\boldsymbol{b}_s \in \mathbb{R}^u$），激活函数为 $\sigma_s(\cdot)$，同样字母“s”为合成 synthesis 的首字母；输出 $\hat{\boldsymbol{x}}$ 为输入 $\boldsymbol{x}$ 的预测估计。注意，这里隐层特征的维数与输入的维数之间的关系为 $u<v$（升维），$u>v$（降维）和 $u=v$（同维）。



3. 优化目标函数

依据不同的损失准则(如能量、熵等)可以构建不同的优化目标函数,下面基于能量的损失构建的优化目标函数为:

$$\min_{\theta} J(\theta)=\frac{1}{N}\sum_{n=1}^{N}\|\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)}\|_2^2+\lambda\cdot R(\theta) \tag{4.3}$$

其中损失项中的输出$\hat{\boldsymbol{x}}^{(n)}$为输入$\boldsymbol{x}^{(n)}$的预测,其期望的输出为$\boldsymbol{x}^{(n)}$,另外参数与正则项定义为:

$$\begin{cases}\theta=[\boldsymbol{W}_a,\boldsymbol{b}_a;\boldsymbol{W}_s,\boldsymbol{b}_s]\\ R(\theta)=\|\boldsymbol{W}_a\|_F^2+\|\boldsymbol{W}_s\|_F^2\end{cases} \tag{4.4}$$

注意超参数中的激活函数$\sigma_a(\cdot)$和$\sigma_s(\cdot)$,以及隐层节点个数(特征维数)事先已给定。

4. 求解

通常,优化目标函数式(4.3)为凸优化问题,可以利用基于随机梯度下降方式的优化迭代算法实现求解:

(1) 合成阶段,目标函数关于参数的偏导数为:

$$\begin{cases}\dfrac{\partial J(\theta)}{\partial \boldsymbol{W}_s}=\dfrac{2}{N}\sum_{n}(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})\cdot\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})^{\mathrm{T}}}{\partial \boldsymbol{W}_s}+2\lambda\dfrac{\partial R(\theta)}{\partial \boldsymbol{W}_s}\\ \dfrac{\partial J(\theta)}{\partial \boldsymbol{b}_s}=\dfrac{2}{N}\sum_{n}(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})\cdot\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})^{\mathrm{T}}}{\partial \boldsymbol{b}_s}\end{cases} \tag{4.5}$$

其中每一样本的误差项(即预测输出与期望输出的差)所对应的偏导数可通过如下的公式求得:

$$\begin{cases}\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})^{\mathrm{T}}}{\partial \boldsymbol{W}_s}=\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)})^{\mathrm{T}}}{\partial \boldsymbol{W}_s}=\sigma_s'\odot\operatorname{diag}(\boldsymbol{X}^{(n)})\in\mathbb{R}^{1\times v}\\ \dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})^{\mathrm{T}}}{\partial \boldsymbol{b}_s}=\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)})^{\mathrm{T}}}{\partial \boldsymbol{b}_s}=\sigma_s'\odot 1_v\in\mathbb{R}^{1\times 1}\end{cases} \tag{4.6}$$

其中$\odot$是向量的点积运算符,表示对应元素相乘;diag(·)是将向量扩展为对角方阵,其对角上的元素为该向量的元素,非对角元素为0;另外1_v为元素为1的v维列向量。σ_s'为合成阶段激活函数的导数。

(2) 分解阶段,目标函数关于参数的偏导数为:

$$\begin{cases}\dfrac{\partial J(\theta)}{\partial \boldsymbol{W}_a}=\dfrac{2}{N}\sum_{n}\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})}{\partial \boldsymbol{W}_a}\cdot(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})^{\mathrm{T}}+2\lambda\dfrac{\partial R(\theta)}{\partial \boldsymbol{W}_a}\\ \dfrac{\partial J(\theta)}{\partial \boldsymbol{b}_a}=\dfrac{2}{N}\sum_{n}\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})}{\partial \boldsymbol{b}_a}\cdot(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})\end{cases} \tag{4.7}$$

为了便于分析,引入误差传播项(即每一样本的误差项关于隐层输出的导数)并记为:

$$\boldsymbol{\delta}_{X^{(n)}}=\left(\frac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})}{\partial X^{(n)}}\right)\in\mathbb{R}^{v\times u} \tag{4.8}$$

进一步，有根据链式法则有：

$$\begin{cases}\dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})}{\partial \boldsymbol{W}_a}=\boldsymbol{\delta}_{X^{(n)}}\cdot\dfrac{\partial X^{(n)}}{\partial \boldsymbol{W}_a}=\boldsymbol{\delta}_{X^{(n)}}\cdot(\boldsymbol{\sigma}_a'\odot\mathrm{diag}(\boldsymbol{x}^n))^{\mathrm{T}}\in\mathbb{R}^{v\times 1}\\ \dfrac{\partial(\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)})}{\partial \boldsymbol{b}_a}=\boldsymbol{\delta}_{X^{(n)}}\cdot\dfrac{\partial X^{(n)}}{\partial \boldsymbol{b}_a}=\boldsymbol{\delta}_{X^{(n)}}\cdot(\boldsymbol{\sigma}_a'\odot 1_u)^{\mathrm{T}}\in\mathbb{R}^{v\times u}\end{cases}\tag{4.9}$$

其中⊙和 diag(·)与合成阶段解释一致，另外 1_u 是元素为 1 的 u 维列向量。σ_a' 为分析阶段激活函数的导数。

(3) 正则项关于参数的偏导数易求，即

$$\begin{cases}\dfrac{\partial R(\theta)}{\partial \boldsymbol{W}_s}=2\boldsymbol{W}_s\\ \dfrac{\partial R(\theta)}{\partial \boldsymbol{W}_a}=2\boldsymbol{W}_a\end{cases}\tag{4.10}$$

基于以上的分析，可得优化更新参数的公式为：

$$\begin{cases}\boldsymbol{W}_a^{(k+1)}=\boldsymbol{W}_a^{(k)}-\alpha\cdot\dfrac{\partial J(\theta)}{\partial \boldsymbol{W}_a}\Big|_{\boldsymbol{W}_a=\boldsymbol{W}_a^{(k)}}\\ \boldsymbol{b}_a^{(k+1)}=\boldsymbol{b}_a^{(k)}-\alpha\cdot\dfrac{\partial J(\theta)}{\partial \boldsymbol{b}_a}\Big|_{\boldsymbol{b}_a=\boldsymbol{b}_a^{(k)}}\\ \boldsymbol{W}_s^{(k+1)}=\boldsymbol{W}_s^{(k)}-\alpha\cdot\dfrac{\partial J(\theta)}{\partial \boldsymbol{W}_s}\Big|_{\boldsymbol{W}_s=\boldsymbol{W}_s^{(k)}}\\ \boldsymbol{b}_s^{(k+1)}=\boldsymbol{b}_s^{(k)}-\alpha\cdot\dfrac{\partial J(\theta)}{\partial \boldsymbol{b}_s}\Big|_{\boldsymbol{b}_s=\boldsymbol{b}_s^{(k)}}\end{cases}\tag{4.11}$$

其中 α 为学习速率。

注意：自编码网络的核心是基于有效的准则建立合理的损失项以期输入与其编码特征(即隐层输出)具有良好的拓扑结构对应性，进一步，其编码特征可以作为新的输入，利用同样的方式(超参数的设置与之前可能有所不同)得到对应的编码特征；依次循环，最终堆栈形成深度神经网络；这里编码特征可以视为输入的一种合理描述，随着层级的增加，编码特征愈加抽象、具有全局整体特性。

4.1.3　自编码网络的常见范式

根据(好的)特征表示衡量标准，常见的自编码网络包括两种类型：一种是编码特征可以较好地重构出输入(例如稀疏自编码、卷积自编码)；二是对输入一定程度下的扰动具有不变性(例如降噪自编码、可收缩性自编码)。下面简述这几种常见的自编码网络。

1. 稀疏自编码

所谓稀疏自编码是指隐层特征具有稀疏响应特性(注意不限制隐层特征的维数一定大于输入信号的维数，这一点可以参考稀疏编码与稀疏表示之间的关系)，其结构如图 4.3 所



示。通常，稀疏性的引入带来的优势有：一是编码方案存储能力大，具有联想记忆能力，并且计算简便；二是使自然信号的结构更加清晰；三是编码方案既符合生物进化普遍的能量最小经济策略，又满足电生理实验的结论。如何引入稀疏性？通常有两种方式，一是不考虑隐层特征的维数与输入维数之间的关系，利用 KL 距离引入稀疏性约束；二是要求隐层特征的维数大于输入的维数，利用伪范数 $L_p(p\in[0,1))$或范数 L_1 正则项引入稀疏性。

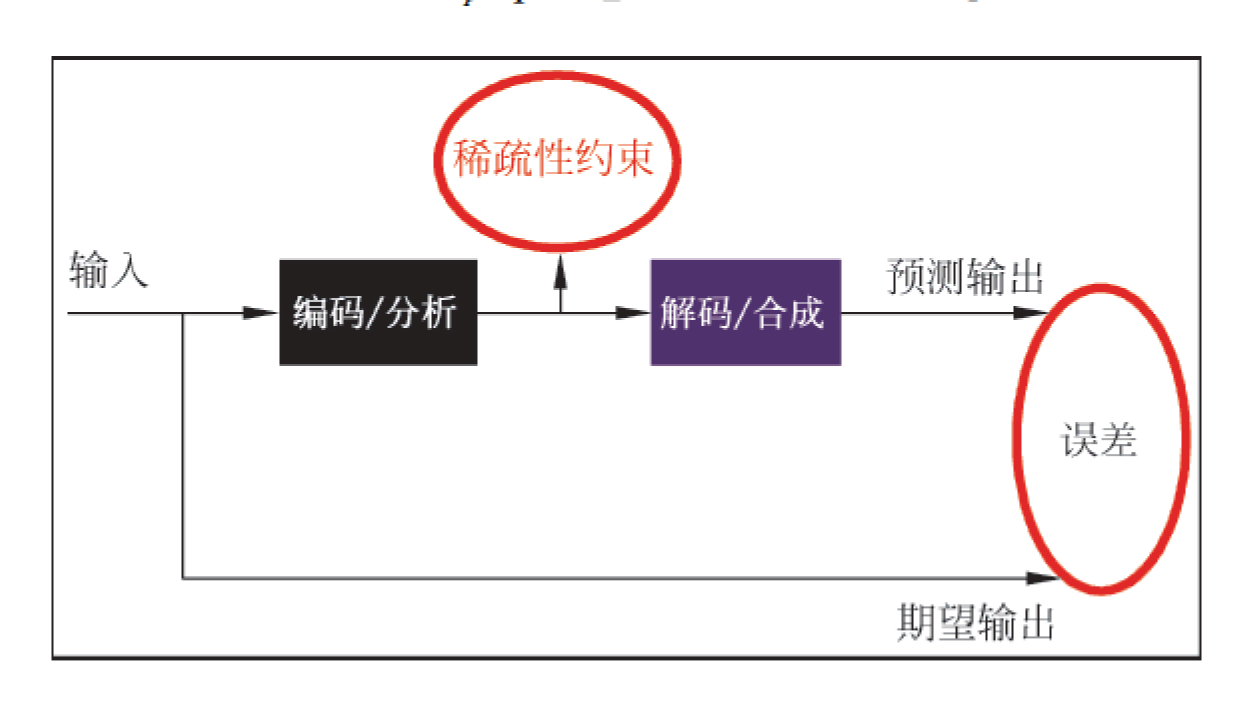


图 4.3　稀疏自编码网络的结构

首先，简述利用 KL 距离引入稀疏性约束：对于数据集（沿用 4.1.2 节的符号表示）有隐层特征的输出：

$$\{\boldsymbol{x}^{(n)}\in\mathbb{R}^{u}\}_{n=1}^{N}\longrightarrow\{\boldsymbol{X}^{(n)}\in\mathbb{R}^{v}\}_{n=1}^{N} \tag{4.12}$$

利用如下公式计算隐层输出每个节点的平均值：

$$\bar{\boldsymbol{X}}=\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{X}^{(n)}\in\mathbb{R}^{v} \tag{4.13}$$

期望隐层每个节点的平均输出值尽量为 0，大部分的隐层节点处于静默状态，为了量化隐层这种特性，通常假设隐层每个节点以一定的概率（小概率发生）进行响应，且节点之间相互独立，注意事先需给出隐层每个节点响应或发生的期望（概率，如 $\rho=0.05$）。进一步，利用 KL 距离构造的稀疏正则项为：

$$\mathrm{KL}(\rho\,\|\,\bar{X}(j))=\rho\cdot\log\left(\frac{\rho}{\bar{X}(j)}\right)+(1-\rho)\cdot\log\left(\frac{1-\rho}{1-\bar{X}(j)}\right) \tag{4.14}$$

其中 $\bar{X}(j)$为 $\bar{\boldsymbol{X}}$ 的第 j 个元素，即隐层第 j 个节点响应的平均值，其中 $j=1,\cdots,v$。在自编码网络优化目标公式(4.3)的基础上，得到稀疏自编码网络的优化目标函数为：

$$\min_{\theta}J(\theta)==\frac{1}{N}\sum_{n=1}^{N}\|\,\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)}\,\|_2^2+\lambda\cdot R(\theta)+\beta\cdot\sum_{j=1}^{v}\mathrm{KL}(\rho\,\|\,\bar{X}(j)) \tag{4.15}$$

优化求解与之前相比，增加稀疏正则项（只与分析阶段的参数$(\boldsymbol{W}_a,\boldsymbol{b}_a)$有关）。其次，考虑利用伪范数 L_p 或范数 L_1 正则项引入稀疏性，对于隐层节点的输出构造的范数约束项为：

$$\begin{cases}\dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\|\,\boldsymbol{X}^{(n)}\,\|_1=\frac{1}{N}\sum_{n=1}^{N}\sum_{j=1}^{v}|\,X^{(n)}(j)\,|\\[2ex]\dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\|\,\boldsymbol{X}^{(n)}\,\|_0\end{cases} \tag{4.16}$$

这里给出了伪范数 L_0 和 L_1 范数的正则项，同样，在自编码网络优化目标函数的基础上，得到稀疏自编码网络的优化目标函数为：

$$\min_{\theta} J(\theta) == \frac{1}{N}\sum_{n=1}^{N}\| \hat{\boldsymbol{x}}^{(n)} - \boldsymbol{x}^{(n)} \|_2^2 + \lambda \cdot R(\theta) + \beta \cdot \frac{1}{N}\sum_{n=1}^{N}\| X^{(n)} \|_1 \tag{4.17}$$

同理基于伪范数 L_0 的稀疏自编码网络的优化目标函数也可以得到，注意求解可以参考稀疏表示中的优化求解算法。

2. 卷积自编码

卷积自编码网络的核心是在层级连接之间引入卷积操作，改变普通自编码网络中的全连接模式，即对于式(4.2)，有：

$$\begin{cases} \boldsymbol{X} = \sigma_a(\boldsymbol{W}_a * \boldsymbol{x} + \boldsymbol{b}_a) \in \mathbb{R}^v \\ \hat{\boldsymbol{x}} = \sigma_s(\boldsymbol{W}_s * \boldsymbol{X} + \boldsymbol{b}_s) \in \mathbb{R}^u \end{cases} \tag{4.18}$$

其中的“ * ”为卷积操作，原来层级间的全连接变为局部连接，例如之前的权值矩阵 $\boldsymbol{W}_a \in \mathbb{R}^{v\times u}$ 参数个数为 $u \cdot v$ 个，但现在，卷积操作下参数的个数减少，例如 Full 卷积操作下 $\boldsymbol{W}_a \in \mathbb{R}^{v-u+1}(v \geqslant u-1)$ 参数个数为 $v-u+1$，以及 Valid 卷积操作下的 $\boldsymbol{W}_a \in \mathbb{R}^{u-v+1}(u \geqslant v-1)$ 参数个数为 $u-v+1$。值得指出的是：卷积自编码网络的输入不再限制为一维向量，对于二维图片也可以进行操作，除了局部连接特性以外，还可以引入权值共享机制(通过层级特征图之间的连接表)，可以参考深度卷积神经网络中的相关知识点。

3. 降噪自编码

为了学习到较为鲁棒的特征，可以对输入数据(在网络中也称可视层)引入随机(加性)噪声，此时对于自编码网络而言，输入为带有噪声的数据，期望输出为没有噪声的数据；需要指出的是：这里带噪声的数据也可以理解为对数据进行某种已知的退化操作所得到的。为什么会带来更为鲁棒性的特征学习？直观地解释为：如人眼在看物体时，如果物体的某一小部分被遮住了，人类依然能够将其识别出来；又如多模态信息输入大脑，少了其中某些模态的信息有时影响也不太大。另外普通自编码网络的缺点是当训练样本与测试样本不符合同一分布时，刻画的特征较差，效果不好。

假设，对于数据集引入随机噪声，即

$$\begin{cases} \{\boldsymbol{x}^{(n)}\}_{n=1}^{N} \xrightarrow{\text{加入噪声}} \{\tilde{\boldsymbol{x}}^{(n)}\}_{n=1}^{N} \\ \tilde{\boldsymbol{x}}^{(n)} = \boldsymbol{x}^{(n)} + \boldsymbol{\varepsilon}^{(n)} \\ \boldsymbol{\varepsilon}^{(n)} \sim \mathrm{E}(\vartheta) \end{cases} \tag{4.19}$$

其中 $\mathrm{E}(\vartheta)$ 为噪声的分布类型，ϑ 为已知的参数。

相应的模型为：

$$\begin{cases} \widetilde{\boldsymbol{X}} = \sigma_a(\boldsymbol{W}_a \cdot \tilde{\boldsymbol{x}} + \boldsymbol{b}_a) \in \mathbb{R}^v \\ \hat{\boldsymbol{x}} = \sigma_s(\boldsymbol{W}_s \cdot \widetilde{\boldsymbol{X}} + \boldsymbol{b}_s) \in \mathbb{R}^u \end{cases} \tag{4.20}$$



其中$\tilde{\boldsymbol{x}}=\boldsymbol{x}+\boldsymbol{\varepsilon}$，进一步，优化目标函数为：

$$\min_{\theta} J(\theta)=\frac{1}{N}\sum_{n=1}^{N}\|\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)}\|_2^2+\lambda\cdot R(\theta) \tag{4.21}$$

注意$\tilde{\boldsymbol{x}}^{(n)}$为输入，$\hat{\boldsymbol{x}}^{(n)}$为输出，期望输出为没有噪声的数据$\boldsymbol{x}^{(n)}$，求解与之前的描述一样。迫使自编码网络去学习输入信号的更加鲁棒的表达，这也是它泛化能力比一般编码器强的原因。

4. 可收缩性自编码

可收缩性自编码网络是自编码网络的一个变种，本质上，它修正了普通自编码网络中的正则项，不直接对层级之间的连接矩阵进行惩罚，而是利用隐层的输出关于输入的雅克比矩阵来进行惩罚。首先给出雅克比矩阵的公式：

$$\boldsymbol{J}_X(x)=\begin{bmatrix}\frac{\partial X(1)}{\partial x(1)} & \frac{\partial X(1)}{\partial x(2)} & \cdots & \frac{\partial X(1)}{\partial x(u)}\\ \frac{\partial X(2)}{\partial x(1)} & \frac{\partial X(2)}{\partial x(2)} & \cdots & \frac{\partial X(2)}{\partial x(u)}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial X(v)}{\partial x(1)} & \frac{\partial X(v)}{\partial x(2)} & \cdots & \frac{\partial X(v)}{\partial x(u)}\end{bmatrix}\in\mathbb{R}^{v\times u} \tag{4.22}$$

其中符号$\boldsymbol{J}_X(x)$为隐层输出X关于输入x的雅克比矩阵，它可以包含数据在各个方向上的信息，对于它的正则性约束可以抑制训练样本在所有方向上的扰动；接下来给出可收缩性自编码器的优化目标函数为：

$$\min_{\theta} J(\theta)=\frac{1}{N}\sum_{n=1}^{N}\|\hat{\boldsymbol{x}}^{(n)}-\boldsymbol{x}^{(n)}\|_2^2+\lambda\cdot\|J_X(\boldsymbol{x})\|_F^2 \tag{4.23}$$

关于求解，其核心在于正则项关于参数的偏导。

4.2 深度堆栈网络

深度堆栈网络是指基于深度前馈神经网络的架构，其中相邻层级之间的(参数)学习策略采用自编码网络来实现，最后将自编码网络中的分析(编码)部分拿出以堆栈形式形成的网络，如图 4.4 所示。

下面针对分类问题，具体说明逐层学习(参数初始化策略)和精调(对深度前馈神经网络整体做端到端的微调)的作用。注意：通常对于分类问题，网络的设计由两部分构成，一部分为特征学习；另一部分为分类器设计。

1. 数据

数据集分为两部分，一部分为有类标的数据集(占整个数据集少部分)，或称为训练集；

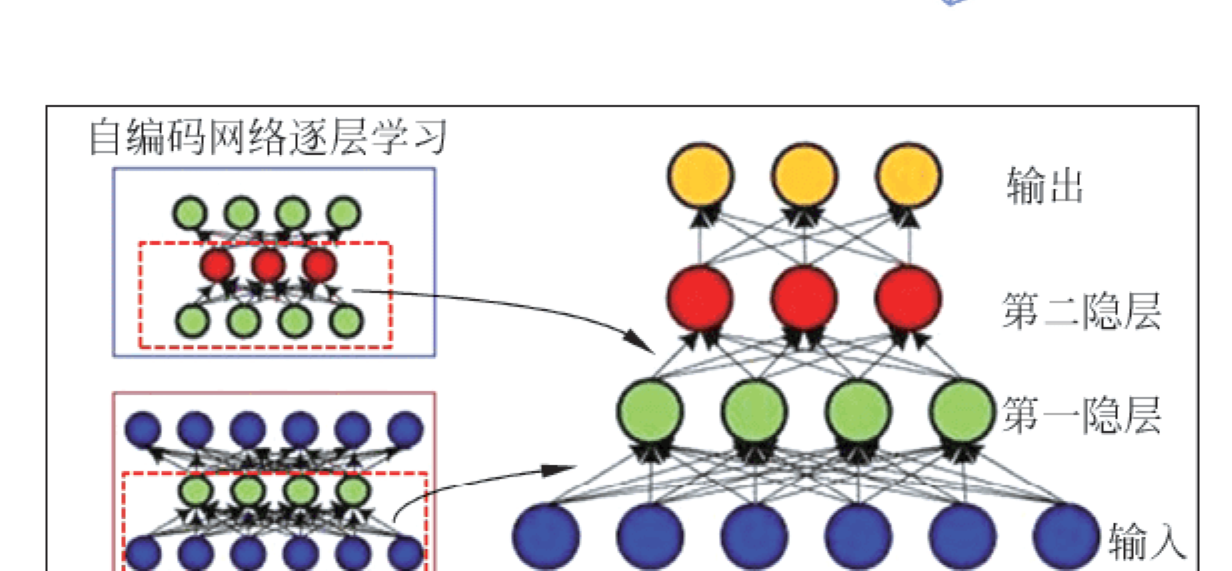


图 4.4　深度堆栈(两个隐层)网络

另一部分为无类标数据集(占整个数据集大部分),或称为测试集,记为:

$$\begin{cases} \{\boldsymbol{x}^{(n)}, y^{(n)}\}_{n=1}^{N} \longrightarrow \text{TrianData} \\ \{\tilde{\boldsymbol{x}}^{(t)}\}_{t=1}^{T} \longrightarrow \text{TestData} \end{cases} \tag{4.24}$$

整个数据集的个数为 $N+T$。

2. 模型

模型的目的:期望在训练集上的性能指标能够在测试集上有同样或相当好的表现,即测试误差尽可能呈现训练误差的下降趋势,提升关于测试集预测准确率的置信度。模型的超参数设置如下:在特征学习阶段具有 L 个隐层,每个隐层上的节点个数为 $n_l(l=1,2,\cdots,L)$,每个隐层上的激活函数为 $\sigma_l(\cdot)$;在分类器设计阶段,选择 Softmax 分类器(当然也可以选择支撑向量机、支撑矩阵机等)。

1) 特征学习阶段

$$\begin{cases} \boldsymbol{X}_l = \sigma_l(\boldsymbol{W}_l \cdot \boldsymbol{X}_{l-1} + \boldsymbol{b}_l) \in \mathbb{R}^{n_l} \\ \boldsymbol{X}_0 = \boldsymbol{x} \end{cases} \tag{4.25}$$

其中 $l=1,2,\cdots,L$。每一层参数初始化的自编码网络模型为:

$$\begin{cases} \boldsymbol{X}_l = \sigma_l^a(\boldsymbol{W}_l^a \cdot \boldsymbol{X}_{l-1} + \boldsymbol{b}_l^a) \\ \hat{\boldsymbol{X}}_{l-1} = \sigma_l^s(\boldsymbol{W}_l^s \cdot \boldsymbol{X}_l + \boldsymbol{b}_l^s) \end{cases} \tag{4.26}$$

其中参数 $\boldsymbol{W}_l=\boldsymbol{W}_l^a$,$\boldsymbol{b}_l=\boldsymbol{b}_l^a$,以及激活函数 $\sigma_l=\sigma_l^a$,角标"a"表示分析阶段,"s"为合成阶段。

2) 分类器设计阶段

假设分类个数为 K,则有:

$$\begin{cases} y(k) = \dfrac{\mathrm{e}^{(X_L \cdot \theta_k)}}{\sum_{j=1}^{K} \mathrm{e}^{(X_L \cdot \theta_j)}} \\ \boldsymbol{y} = [y(1), y(2), \cdots, y(K)]^{\mathrm{T}} \end{cases} \tag{4.27}$$

待学习的参数为 $\theta_k(k=1,2,\cdots,K)$。



3. 优化目标函数

依据参数初始化和精调两个阶段来构造优化目标函数。

1）参数初始化化阶段(特征学习阶段)

$$\min_{(W_l^{(a)},b_l^{(a)},W_l^{(s)},b_l^{(s)})} \frac{1}{T}\sum_{t=1}^{T} \| \hat{X}_{l-1}^{(t)} - X_{l-1}^{(t)} \|_2^2 + \lambda \cdot (\| W_l^{(a)} \|_F^2 + \| W_l^{(s)} \|_F^2) \quad (4.28)$$

根据式(4.26)在无类标数据集或测试集上得到该优化目标函数(参数初始化的学习方式为无监督形式)。其中 $l=1,2,\cdots,L$，以及优化求解后令分析阶段的参数为 $W_l=W_l^{(a)}$，$b_l=b_l^{(a)}$。

2）精调(特征学习＋分类器设计)

$$\min_{(W,b;\theta)} J(W,b;\theta) = \frac{1}{N}\sum_{n=1}^{N}\mathrm{loss}(\hat{y}^{(n)},y^{(n)}) + \lambda \cdot R(W) + \beta \cdot R(\theta) \quad (4.29)$$

其中损失函数可以利用交叉熵来构造，即

$$\mathrm{loss}(\hat{y},y) = -\sum_{k=1}^{K}\delta(y(k)=1)\cdot \log(\hat{y}(k)) \quad (4.30)$$

注意对于样本(x,y)，输出目标 y 为 K 维的 one-hot 向量，如对于输入 x，其类标为第 k 类，则：

$$y = (0 \quad \cdots \quad \underset{\underset{k}{\uparrow}}{1} \quad \cdots \quad 0)^{\mathrm{T}} \in \mathbb{R}^K$$

另外 $\delta(\cdot)$为示性函数，即 $y(k)=1$ 成立时为 1，否则为 0；另外正则项约束有：

$$\begin{cases} R(W) = \sum_{l=1}^{W} \| W_l \|_F^2 \\ R(\theta) = \sum_{k=1}^{K} \| \theta_k \|_F^2 \end{cases} \quad (4.31)$$

需要注意的是精调在有类标数据集或训练集上完成。

4. 求解

优化求解的目标函数为式(4.28)和式(4.29)，其中前者使用浅层网络(通常，对应凸优化算法)易得到特征学习阶段的初始化参数，记为：

$$\begin{cases} W^* = \{W_1,W_2,\cdots,W_L\} \\ b^* = \{b_1,b_2,\cdots,b_L\} \end{cases}$$

进一步，初始化分类器设计阶段的参数，记为：

$$\theta^{(0)} = (\theta_1^{(0)},\theta_2^{(0)},\cdots,\theta_K^{(0)})$$

端到端方式精调后的深度堆栈网络的参数为：

$$(W^*,b^*;\theta^{(0)}) \xrightarrow{\text{若干次迭代}} (W^*+\varepsilon,b^*+\varepsilon;\theta^*) \quad (4.32)$$

接下来提出进一步提升深度堆栈网络(也适用于深度学习模型)性能的方法，包括如下四个

方面，一是通过数据提升性能，通常数据的规模与质量决定了网络模型的性能，如何获取更多的数据？基于统计方式的裁剪、旋转、伸缩等扩展数据，或利用数据生成技术（例如生成式对抗网络），另外比较重要的是利用稀疏筛选技术剔除那些离群的数据；二是通过算法提升性能，从算法角度，深度学习模型关于超参数的选择、反向传播算法的计算可以利用经典的机器学习算法辅助，例如利用支撑向量机来研究随着层级的增加，相对应隐层特征的拓扑结构特性的变化从而选择合适的层数等；三是通过端到端方式的精调提升性能，通常的技巧包括权值初始化、学习率、激活函数、网络拓扑、随机批量个数和迭代次数、正则化、早停等；四是通过整合不同功能模块提升性能，整合形式包括模型整合、视角整合和堆栈整合。图4.5为深度堆栈（两个隐层）网络的核心与常用技巧。

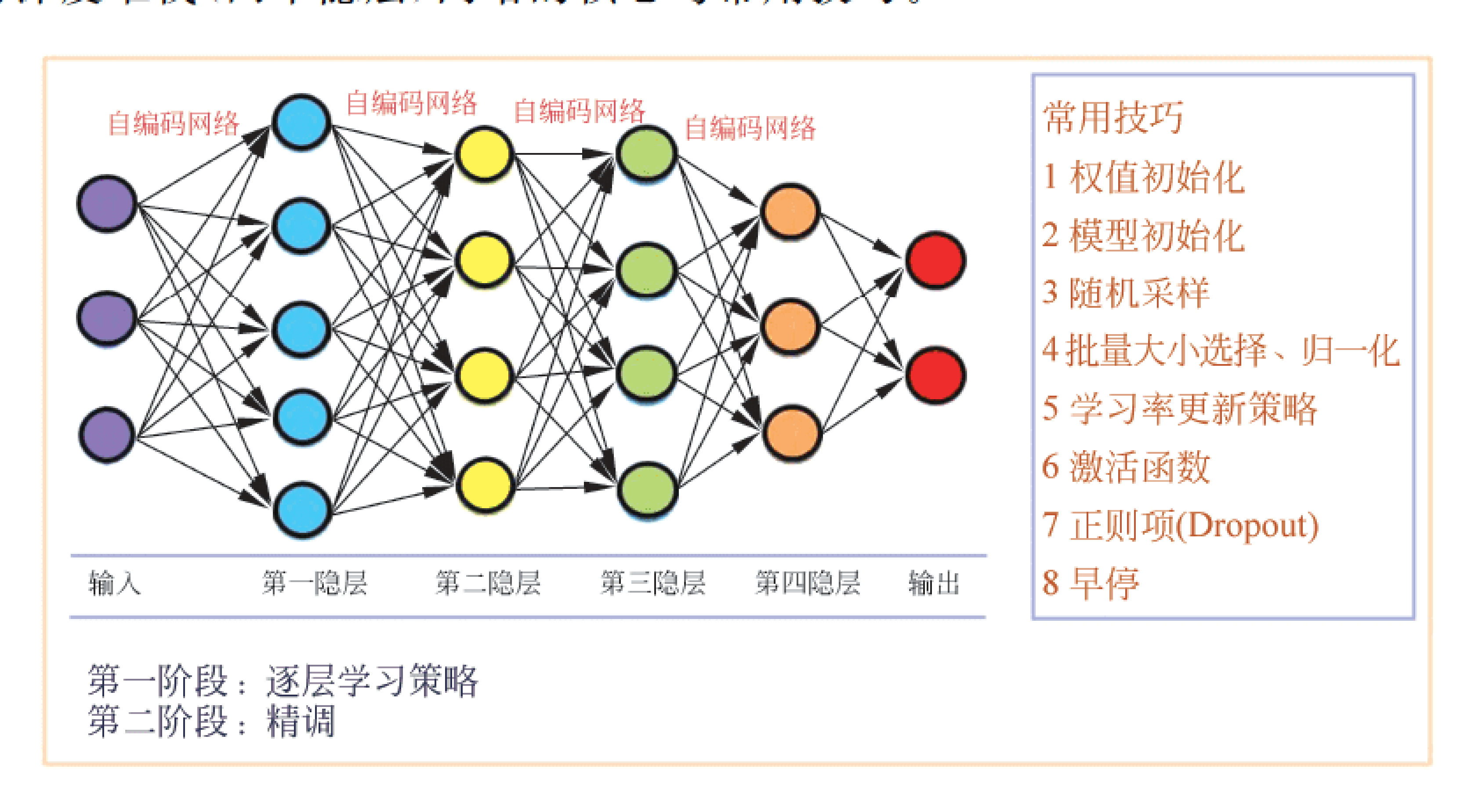


图4.5　深度堆栈（两个隐层）网络的核心与常用技巧

4.3　深度置信网络/深度玻尔兹曼机网络

4.3.1　玻尔兹曼机/受限玻尔兹曼机

关于玻尔兹曼机与受限玻尔兹曼机在第1章中的神经网络部分已经给出严格的数学刻画，本质上，也属于自编码网络的范畴。众所周知，玻尔兹曼机是由随机神经元全连接组成的反馈神经网络，且对称连接，无自反馈，包含两层，一个可视层和一个隐层，层内连接，且层级之间也全连接；而受限玻尔兹曼机是层内无连接，层与层之间全连接，见图4.6。

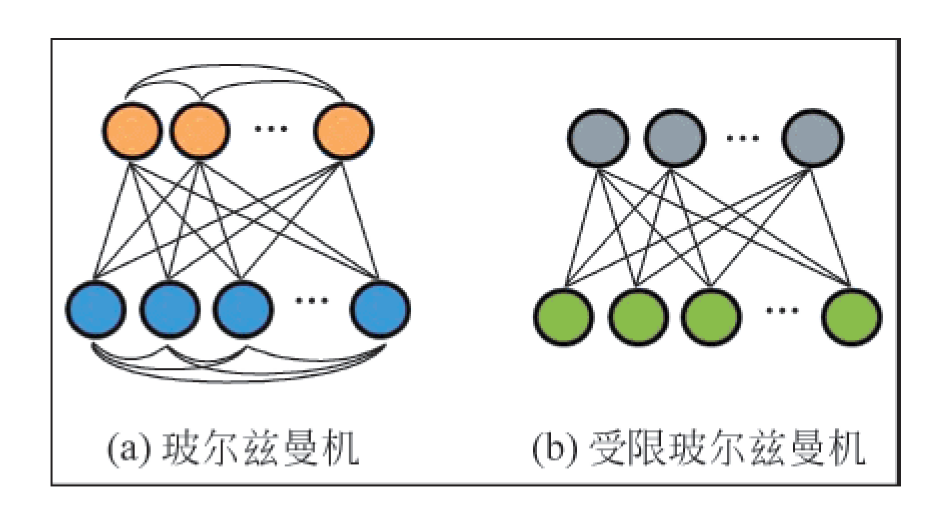


图4.6　玻尔兹曼机与受限玻尔兹曼机的网络结构

由于波尔兹曼机具有很强大的无监督学习能力，能够学习数据中复杂的规则，但代价是训练

(学习)时间很长,此外,还难以准确地计算玻尔兹曼机所表示的分布,进一步获取该分布下的随机样本也很困难,于是引入一种限制玻尔兹曼机。受限玻尔兹曼机网络结构的特点是:在给定可视层单元状态(输入数据)时,各隐层单元的激活条件独立,反过来看,在给定隐层单元状态时,可见层单元的激活条件也是独立的。这样,尽管受限玻尔兹曼机所表示的分布仍无法有效计算,但可通过 Gibbs 采样得到服从该分布的随机样本。只要隐层单元的数目足够,受限玻尔兹曼机就能拟合任意离散分布。为了有效地求解优化目标函数,Hinton 等人于 2002 年提出了一个快速学习算法,即对比散度算法。另外,在应用方面,该模型已经成功被用来解决不同的机器学习问题,比如分类、回归、降维、高维时间序列建模、特征提取等。

4.3.2 深度玻尔兹曼机/深度置信网络

可以认为,深度堆栈网络的核心是逐层预训练与精调,也是目前半监督学习的主流,不论是基于能量(构建损失函数)的自编码网络,还是基于信息量(构建损失函数)的受限波尔兹曼机或波尔兹曼机,针对这一类网络模型的改进与应用十分广泛,下面主要基于受限玻尔兹曼机来详述深度置信网络的拓扑结构与训练技巧。首先,网络的整体结构为深度前馈神经网络,参考图 4.1,只是层级之间的参数初始化利用受限玻尔兹曼机的学习方式获取,即将受限玻尔兹曼机中的(隐层)乘性偏置和权值连接矩阵直接赋给相应层级的权值矩阵和偏置。其次从数据、模型、优化目标函数和求解四个方面来对深度置信网络进行分析与理解。

1. 数据

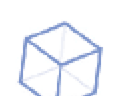
与(基于三层前馈神经网络)自编码网络中的数据要求一致,数据分为有类标数据集和无类标数据集,即式(4.24)。

2. 模型

模型的网络结构为深度前馈神经网络,其模块设计仍分为两个阶段,一是特征学习阶段,即式(4.25);另一个是分类器设计阶段,即式(4.27);与传统自编码网络形成的深度堆栈网络的唯一区别在于特征学习阶段中,层级间的参数初始化网络模型,即

$$\begin{cases} \boldsymbol{h} \sim P(\boldsymbol{h} \mid \boldsymbol{v},\boldsymbol{W},\boldsymbol{a},\boldsymbol{b}) = \dfrac{1}{P(\boldsymbol{v}) \cdot Z} \cdot \mathrm{e}^{(\boldsymbol{b}^{\mathrm{T}} \cdot \boldsymbol{h} + \boldsymbol{v}^{\mathrm{T}} \cdot \boldsymbol{W} \cdot \boldsymbol{h})} \\ \hat{\boldsymbol{v}} \sim \boldsymbol{P}(\boldsymbol{v} \mid \boldsymbol{h},\boldsymbol{W},\boldsymbol{a},\boldsymbol{b}) = \dfrac{1}{P(\boldsymbol{h}) \cdot Z} \cdot \mathrm{e}^{(\boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{v} + \boldsymbol{v}^{\mathrm{T}} \cdot \boldsymbol{W} \cdot \boldsymbol{h})} \end{cases} \tag{4.33}$$

其中 $\boldsymbol{v}$ 为可视层,即输入进行归一化后的数据;$\boldsymbol{h}$ 为隐层,$\boldsymbol{W}$ 为(输入)可视层 $\boldsymbol{v}$ 到隐层间的权值连接矩阵,其转置 $\boldsymbol{W}^{\mathrm{T}}$ 为隐层到(输出)可视层 $\hat{\boldsymbol{v}}$ 的权值连接矩阵,符号 $\boldsymbol{a}$ 为可视层上的乘性偏置,符号 $\boldsymbol{b}$ 为隐层上的乘性偏置;另外 $\boldsymbol{h}$ 服从相应的分布,通过吉布斯采样获取。但在实际应用中,具体的获取方式为:若已知参数$(\boldsymbol{W},\boldsymbol{a},\boldsymbol{b})$,根据输入 $\boldsymbol{v}$,隐层 $\boldsymbol{h}$ 的计算公式为:



$$\begin{cases} P(h(i)=1 \mid \boldsymbol{v}) = \sigma(\boldsymbol{v}^{\mathrm{T}} \cdot \boldsymbol{W}_{:,i} + b_i) \\ P(h(i)=0 \mid \boldsymbol{v}) = 1-\sigma(\boldsymbol{v}^{\mathrm{T}} \cdot \boldsymbol{W}_{:,i} + b_i) \end{cases} \tag{4.34}$$

其中的$\sigma(\cdot)$为 Sigmoid 函数，其中 $h(i)$为隐层 $\boldsymbol{h}$ 中第 i 个节点的输出值，$\boldsymbol{W}_{:,i}$为权值矩阵 $\boldsymbol{W}$ 的第 i 列，b_i 为隐层乘性偏置 $\boldsymbol{b}$ 的第 i 个成分；同理根据隐层 $\boldsymbol{h}$，对可视层进行估计的公式为：

$$\begin{cases} P(v(j)=1 \mid \boldsymbol{h}) = \sigma(\boldsymbol{W}_{j,:} \cdot h + a_j) \\ P(v(j)=0 \mid \boldsymbol{h}) = 1-\sigma(\boldsymbol{W}_{j,:} \cdot h + a_j) \end{cases} \tag{4.35}$$

其参数的解释与上述一致，这里不再赘述。

备注：本小节出现的公式及推理，具体解释可参考第 1 章。

3. 优化目标函数

优化目标函数分为两个阶段，一是参数初始化；二是精调（与式(4.29)一致）；根据式(4.33)，关于层级参数初始化的优化目标函数为：

$$\min_{\theta} J(\theta) = -\sum_{t=1}^{T} \log P(\hat{\boldsymbol{v}}^{(t)}) = -\sum_{t=1}^{T} \log \sum_{h} P(\hat{\boldsymbol{v}}^{(t)}, \boldsymbol{h}) \tag{4.36}$$

例如，输入与第一隐层之间的参数初始化解释如下，其中将数据归一化后得到可视层：

$$\{\tilde{\boldsymbol{x}}^{(t)}\}_{t=1}^{T} \xrightarrow{\text{归一化}} \{\boldsymbol{v}^{(t)}\}_{t=1}^{T}$$

利用式(4.36)得到参数为：

$$(\boldsymbol{W}^{*}, \boldsymbol{a}^{*}, \boldsymbol{b}^{*})$$

并将其赋给深度前馈神经网络中的第一层之间的权值矩阵与（加性）偏置，即

$$\begin{cases} \boldsymbol{W}_1 = \boldsymbol{W}^{*} \\ \boldsymbol{b}_1 = \boldsymbol{b}^{*} \end{cases} \tag{4.37}$$

接下来，第一隐层到第二隐层之间的参数初始化解释如下：先利用如下公式得到深度前馈神经网络中第一隐层的输出：

$$\boldsymbol{X}_1 = \sigma_1(\boldsymbol{W}_1 \cdot \boldsymbol{x} + \boldsymbol{b}_1) \tag{4.38}$$

其中 $\sigma_1(\cdot)$为第一隐层上的激活函数，进而数据在进行归一化后得：

$$\{\boldsymbol{X}_1^{(t)}\}_{t=1}^{\mathrm{T}} \xrightarrow{\text{归一化}} \{\boldsymbol{v}_1^{(t)}\}_{t=1}^{\mathrm{T}}$$

同样，利用式(4.36)得到参数（注意此时输入发生变化，相应参数的尺寸也随之变化），得到的参数赋给$(\boldsymbol{W}_2, \boldsymbol{b}_2)$；以此进行，可以将特征学习阶段的每一层参数训练完毕，即完成参数初始化的过程。

4. 求解

关于精调阶段的求解，与传统的深度前馈神经网络一致，利用反向传播算法进行端到端网络整体微调。参数初始化阶段的优化目标函数(4.36)利用对比散度算法求解（一种逼近

算法)。

另外,基于玻尔兹曼机结构的深度玻尔兹曼机网络,与深度置信网络类似,这里不再赘述。综上所述,深度置信网络的结构图如图 4.7 所示。

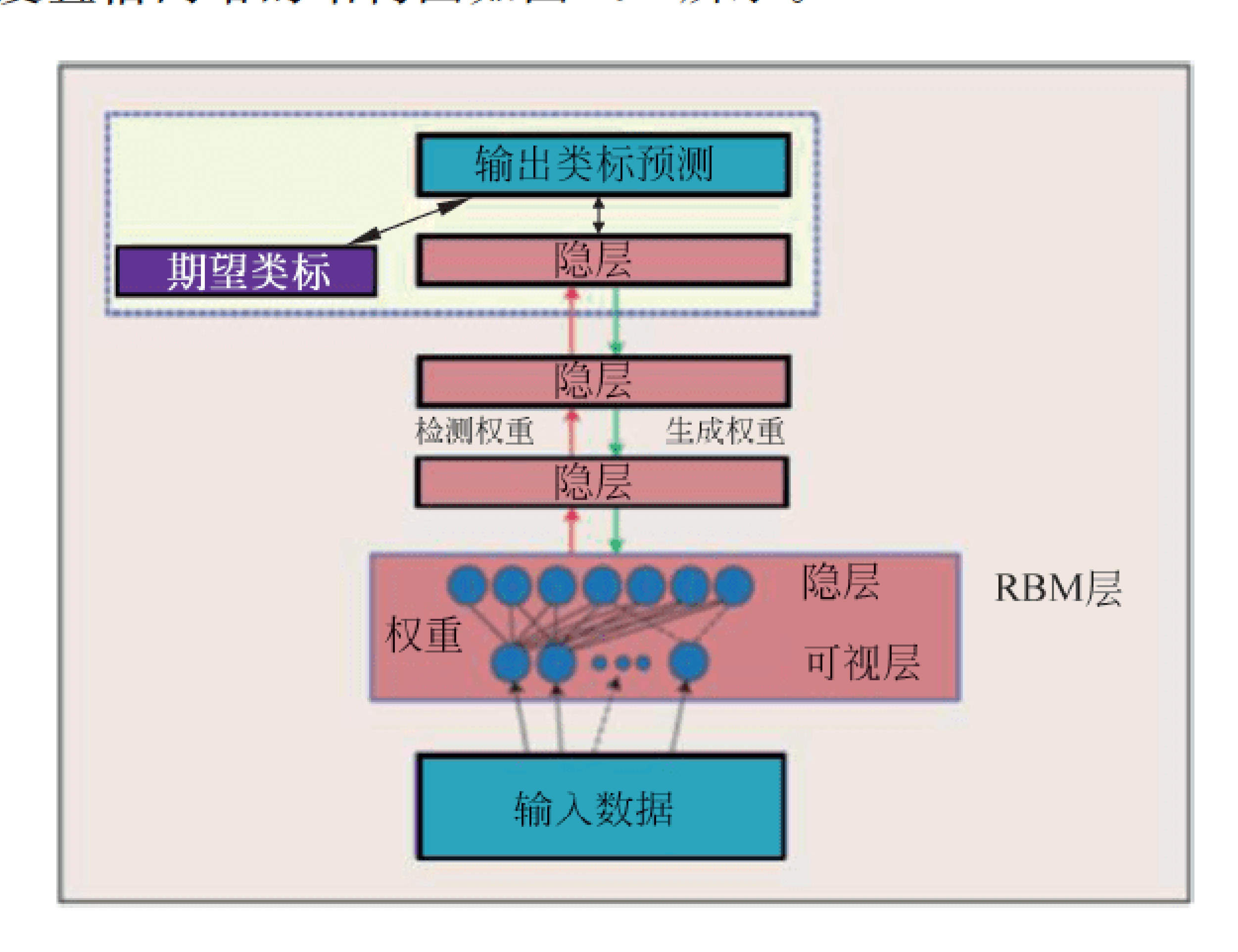


图 4.7 深度置信网络的结构图

备注:目前,深度置信网络已经成功应用于模式分类等问题中,但仍有大量工作或问题需要解决。例如急需解决以下三个问题:一是理论方面,其包括两个方面,一个是数学物理层面的,另一个是计算方面的。众所周知,深度模型相比较于浅层模型有更好的(对非线性函数的)表征能力,但对于某些类函数,深度网络仅仅需要非常少的参数就可以表示。需要指出的是可表示性不代表可学习性;还有需要多少训练样本才能学习到足够好的深度模型。另一方面,需要多少计算资源才能训练出更好的模型。二是建模问题,自从深度信念网络提出后,产生了许多类似的结构,如用自动编码器替代受限波尔兹曼机,能否提出新的分层模型(例如稀疏层次目标识别,即 Sparse-Heirarchical MAX,S-HMAX),使其不但有传统深度模型所具有的强大表示能力,而且更容易做理论分析。三是并行优化问题,深度信念网络中的反向传播算法,基于最小批处理的随机梯度优化算法很难在多计算机中进行并行训练。通常使用 GPU 加速,然而单个机器 GPU 对大规模数据识别或相似任务数据集并不适用。未来,云计算平台和基于 FPGA 的并行加速,是深度信念网络在大规模数据识别的重点研究问题。

参考文献

[4.1] Vincent P, Larochelle H, Bengio Y, et al. Extracting and composing robust features with denoising autoencoders[C]. International Conference. 2008: 1096-1103.

[4.2] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks[J]. Science, 2006, 313(5786): 504.

[4.3] Bengio Y, Lamblin P, Popovici D, et al. Greedy layer-wise training of deep networks[C]. International Conference on Neural Information Processing Systems. MIT Press, 2006: 153-160.

[4.4] Vincent P, Larochelle H, Lajoie I, et al. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion[J]. Journal of Machine Learning Research, 2010, 11(12): 3371-3408.

[4.5] Masci J, Meier U, An D, et al. Stacked convolutional auto-encoders for hierarchical feature extraction[C]. International Conference on Artificial Neural Networks. Springer-Verlag, 2011: 52-59.

[4.6] Rifai S, Mesnil G, Vincent P, et al. Higher Order Contractive Auto-Encoder[M]. Machine Learning and Knowledge Discovery in Databases. Springer Berlin Heidelberg, 2011: 645-660.

[4.7] Lange S, Riedmiller M. Deep Auto-Encoder Neural Networks in Reinforcement Learning[J]. 2010: 1-8.

[4.8] Rifai S, Vincent P, Muller X, et al. Contractive Auto-Encoders: Explicit Invariance During Feature Extraction[C]. ICML. 2011.

[4.9] Hinton G E, Krizhevsky A, Wang S D. Transforming Auto-Encoders[C]. Artificial Neural Networks and Machine Learning-ICANN 2011-, International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings. 2011: 44-51.

[4.10] Bengio Y, Yao L, Alain G, et al. Generalized Denoising Auto-Encoders as Generative Models[J]. Advances in Neural Information Processing Systems, 2013: 899-907.

[4.11] Kingma D P, Welling M. Auto-Encoding Variational Bayes[C]. Conference proceedings: papers accepted to the International Conference on Learning Representations (ICLR2014). arXiv.org, 2014.

[4.12] Rifai S, Bengio Y, Dauphin Y, et al. A Generative Process for Sampling Contractive Auto-Encoders[J]. Computer Science, 2012, 2.

[4.13] Kan M, Shan S, Chang H, et al. Stacked Progressive Auto-Encoders (SPAE) for Face Recognition Across Poses[C]. Computer Vision and Pattern Recognition. IEEE, 2014: 1883-1890.

[4.14] Li J, Luong M T, Dan J. A Hierarchical Neural Autoencoder for Paragraphs and Documents[J]. Computer Science, 2015.

[4.15] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18(7): 1527.

[4.16] Hinton G E. Training products of experts by minimizing contrastive divergence[J]. Neural Computation, 2002, 14(8): 1771-1800.

[4.17] Tao C, Pan H, Li Y, et al. Unsupervised Spectral-Spatial Feature Learning With Stacked Sparse Autoencoder for Hyperspectral Imagery Classification[J]. IEEE Geoscience & Remote Sensing Letters, 2015, 12(12): 2438-2442.

[4.18] Bengio Y, Courville A, Vincent P. Representation learning: a review and new perspectives[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2013, 35(8): 1798.

[4.19] Tieleman T. Training restricted Boltzmann machines using approximations to the likelihood gradient[C]. International Conference. DBLP, 2008: 1064-1071.

[4.20] Lee H, Ekanadham C, Ng A Y. Sparse deep belief net model for visual area V2[J]. Advances in Neural Information Processing Systems, 2007, Vol 20: 873-880.

[4.21] Lee H, Grosse R, Ranganath R, et al. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations[C]. International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June. DBLP, 2009: 609-616.

[4.22] Sarikaya R, Hinton G E, Deoras A. Application of Deep Belief Networks for Natural Language Understanding[J]. IEEE/ACM Transactions on Audio Speech & Language Processing, 2014, 22(4): 778-784.

[4.23] Ranzato M, Boureau Y L, Lecun Y. Sparse feature learning for deep belief networks[C]. Advances in Neural Information Processing Systems. 2007: 1185-1192.

[4.24] Roux N L, Bengio Y. Representational Power of Restricted Boltzmann Machines and Deep Belief Networks[J]. Neural Computation, 2008, 20(6): 1631-1649.

[4.25] Salakhutdinov R, Hinton G. Deep Boltzmann Machines[J]. Journal of Machine Learning Research, 2009, 5(2): 1967-2006.

[4.26] Srivastava N, Salakhutdinov R. Multimodal Learning with Deep Boltzmann Machines[J]. Journal of Machine Learning Research, 2012, 15(8): 1967-2006.

[4.27] Salakhutdinov R, Hinton G. An efficient learning procedure for deep Boltzmann machines[J]. Neural Computation, 2012, 24(8): 1967.

[4.28] Salakhutdinov R, Hinton G. A better way to pretrain Deep Boltzmann Machines[C]. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012: 2447-2455.

[4.29] Srivastava N, Salakhutdinov R R, Hinton G E. Modeling Documents with Deep Boltzmann Machines[J]. Computer Science, 2013.

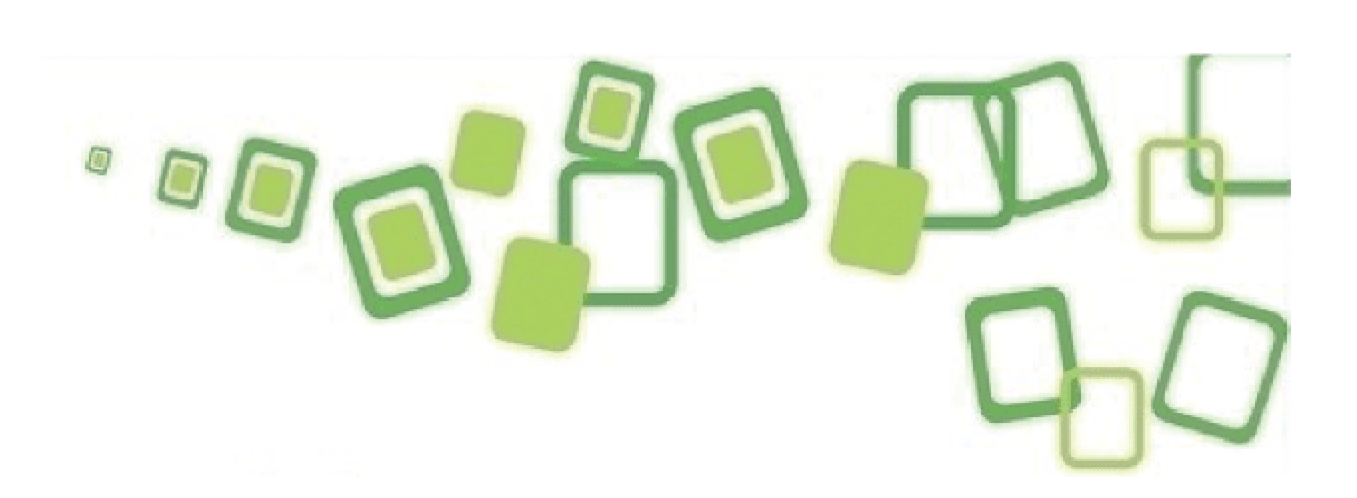
[4.30] Marc, Hinton G E. Modeling pixel means and covariances using factorized third-order boltzmann machines[C]. Computer Vision and Pattern Recognition. IEEE, 2010: 2551-2558.

[4.31] Goodfellow I J, Mirza M, Courville A, et al. Multi-prediction deep Boltzmann machines[C]. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2013: 548-556.

[4.32] Tang Y, Salakhutdinov R, Hinton G. Robust Boltzmann Machines for recognition and denoising[C]. Computer Vision and Pattern Recognition. IEEE, 2012: 2264-2271.

[4.33] Cho K H, Raiko T, Ilin A. Gaussian-Bernoulli deep Boltzmann machine[C]. International Joint Conference on Neural Networks. IEEE, 2013: 1-7.

第5章

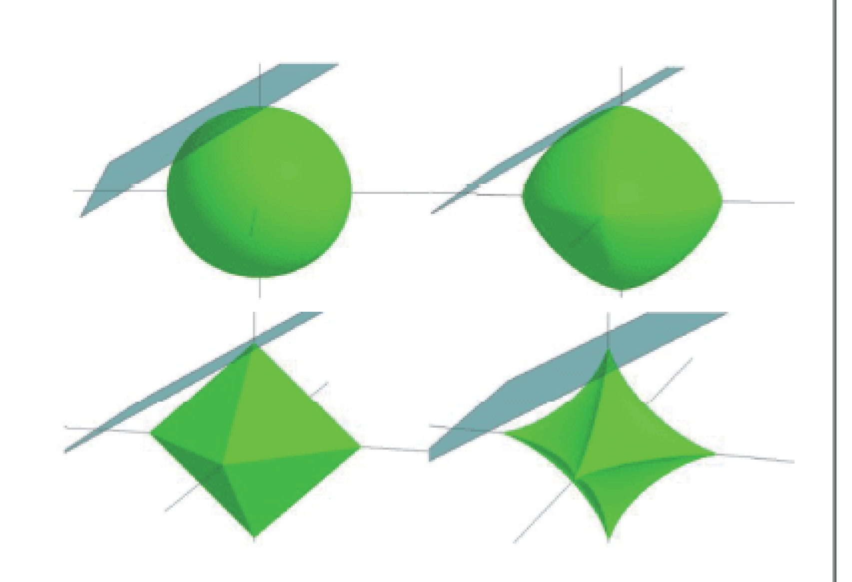


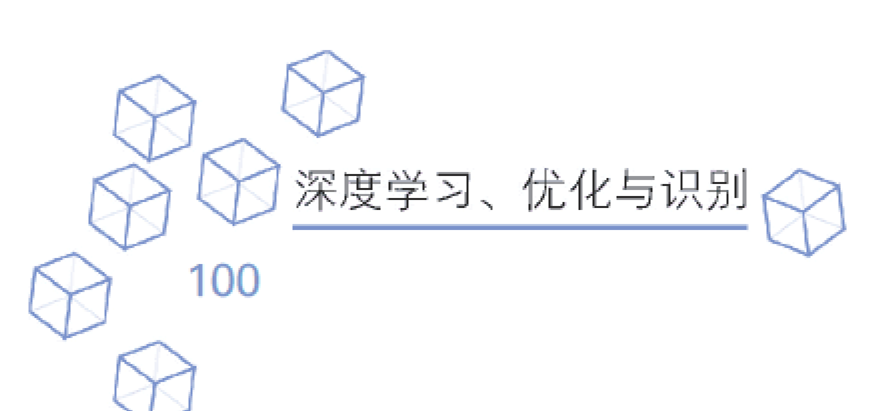
稀疏深度神经网络

CHAPTER 5

深度神经网络中的稀疏表达

- 数据中的稀疏性——稀疏表示假设
- 权值连接矩阵的稀疏性——稀疏技巧
- 激活函数的稀疏性——内蕴稀疏性
- 隐层输出的稀疏性——稀疏正则
- 稀疏模式识别——稀疏分类器设计
- 稀疏初始化策略——稀疏编码、自编码网络





5.1 稀疏性的生物机理

深度学习与稀疏认知学习、计算与识别之间的关系深刻而且本质，从机器学习中的特征工程(人工特征提取与特征筛选)到深度学习中的特征学习(通过线性与非线性操作的不断复合获取数据的高层统计或语义特性)，无论是以显性还是隐性的嵌入方式，稀疏性都在模型中扮演着重要的角色。下面简要从生物视觉机理和数学物理角度来描述稀疏性。

备注：稀疏认知学习、计算与识别的范畴包括表示理论(即基于稀疏表示的压缩感知和稀疏编码)，数学计算(最优匹配追踪算法)和模式识别(稀疏表示分类器 SRC 及稀疏分类器设计 SparseMax)等。

5.1.1 生物视觉机理

视觉感知机理的研究表明，视觉系统可以看成一种合理而且高效的图像处理系统，从视网膜到大脑皮层存在一系列具有不同生物学功能的神经细胞，例如随着层级信息不断的“加深”，不同视觉皮层上的神经细胞对特定形状的视觉图案有最佳的响应和偏好的刺激，简言之，层级越高感受野越大，即信息处理从局部到更大的区域，类似尺度特性。层级较低时，感受野所处理的区域越小，稀疏性越强(特指层级间的连接特性)，层级较高时，感受野所处理的区域越大，稀疏性越弱。另外，Barlow 推论出在稀疏性和自然环境的统计特性之间必然存在某种联系，随后诸多基于生物视觉和计算的模型被提出来，都成功地例证了生物视觉针对自然环境所反馈出的物理统计特性蕴含着稀疏性。当层级较低时，其简单细胞对应着严格的方向和带通特性，而复杂细胞在保持简单细胞特性的基础上进一步具有局部变换(如平移)不变性，简言之，简单细胞处理信息具有稀疏(即局部连接)特性，而复杂细胞具有聚类(连接计算共享)特性。神经科学研究成果表明，稀疏编码是视觉系统中图像表示的主要方式，初级视觉皮层(V1 区)中的神经元对视觉信息的反应具有稀疏性，V4 区的神经元通过稀疏编码的方式实现视觉信息的表示。从表 5.1 中可知，随着对计算机视觉研究的深入，人类对自身视觉感知系统的理解也在不断加深。借鉴生物视觉机理的研究成果，模拟建立相应的视觉计算模型，将成为一个极具挑战性和吸引力的研究方向。下面给出生物(人类)视觉与计算机视觉的对比表(表 5.1)。

表 5.1 生物(人类)视觉与计算机视觉对比

对 比 项	人 类 视 觉	计算机视觉
适应性	适应性强，可在复杂及变化的环境中识别目标	适应性差，容易受复杂背景及环境变化的影响
智能	具有高级智能，可运用逻辑分析及推理能力识别变化的目标，并能总结规律	虽然可利用人工智能及神经网络技术，但智能很差，不能很好地识别变化的目标

续表

对　比　项	人 类 视 觉	计算机视觉
彩色识别能力	对色彩的分辨能力强,但容易受人的心理影响,不能量化	受硬件条件的制约,目前一般的图像采集系统对色彩的分辨能力较差,但具有可量化的优点
灰度分辨能力	差,一般只能分辨 64 个灰度级	强,目前一般使用 256 灰度级,采集系统可具有 10bit、12bit、16bit 等灰度级
空间分辨能力	分辨率较差,不能观看微小的目标	目前有 4K×4K 的面阵摄像机和 8K 的线阵摄像机,通过备置各种光学镜头,可以观测小到微米大到天体的目标
速度	0.1 秒的视觉暂留使人眼无法看清较快速运动的目标	快门时间可达到 10 微秒左右,高速相机帧率可达到 1000 以上,处理器的速度越来越快
感光范围	400～750nm 范围的可见光	从紫外到红外的较宽光谱范围,另外有 X 光等特殊摄像机
环境要求	对环境温度、湿度的适应性差,另外有许多场合对人有损害	对环境适应性强,另外可加防护装置
观测精度	精度低,无法量化	精度高,可到微米级,易量化
其他	主观性,受心理影响,易疲劳	客观性,可连续工作

另外,关于生物视觉与计算机视觉之间核心的模块对应关系见图 5.1,值得注意的是:理解并分析大脑是如何在算法层面上工作的尝试是鲜活且发展良好的,这项尝试被称为"计算神经科学",并且是独立于深度学习的一个领域。研究人员两个领域间反复研究是很常见的,深度学习主要关注如何构建智能的计算机系统,以用来解决需要智能才能解决的任务,而计算神经科学领域主要是关注构建大脑如何工作的更精确的模型。

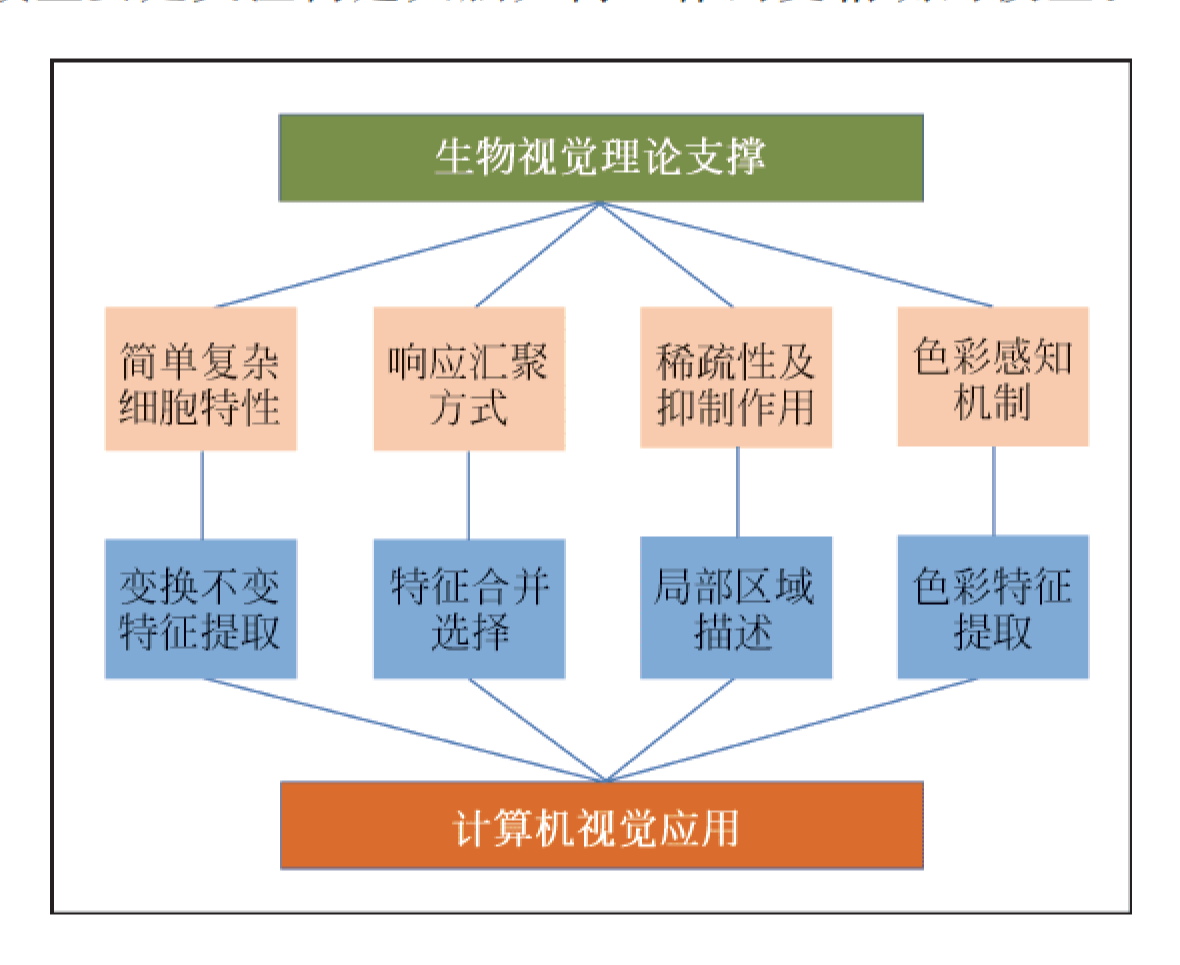


图 5.1　生物视觉与计算机视觉核心模块对应



5.1.2 稀疏性响应与数学物理描述

目前，构建高性能的计算模型，并不是模型越复杂越好，特别针对变量维数很高，样本量不是很大的情形下，构建一个合理的，相对简单的稀疏模型往往具有更高的性能，更为重要的是还具有生物可解释性。从数学角度来看，依据模型的低复杂性结构(如向量的稀疏性，矩阵的低秩性等)，如何高效地从病态的线性逆问题中唯一且稳健地恢复出特定的信息。值得指出的是：常见的稀疏性是指向量中绝大多数元素的值为零或者接近于零；而广义的稀疏性是指通过特定变换后目标的稀疏性。可以看出，当前为了使得模型具备学习能力、高容量的表达能力、快速推断能力以及多任务信息共享能力；借鉴生物视觉的认知机理已成为一种必然趋势。众所周知，1996 年 Olshausen 和 Field 在 Nature 杂志上发表的一篇重要论文指出，自然图像经过稀疏编码后得到的基函数类似于初级视觉皮层 V1 区上简单细胞感受野的反应特性(即空间域的局部性、时域和频域的方向性和选择性)。需要指出的是稀疏编码与稀疏表示是不同的，例如关于系数的稀疏性约束，前者采用光滑可导的函数，而后者采用伪范数或 L1 范数；另外稀疏编码不要求基原子个数一定要大于数据的维数。本节更为详细的论述与解释请参考第 1 章的稀疏表示，另外稀疏编码部分可参考相关论文，这里不再赘述。

5.2 稀疏深度网络模型及基本性质

在深度神经网络引入显式稀疏性之前，关于稀疏模型的研究就已经成为机器学习中的热点，特别是针对线性稀疏模型的研究，如压缩感知、双稀疏模型、结构化稀疏模型(如群稀疏)、S-HMAX 模型、SRC 模型等。当然，除了显式稀疏性(如稀疏正则化理论等)外，还有隐式稀疏性的研究，它通常内蕴在非线性激活函数和损失函数(如交互熵，非 L2 范数下的能量损失)的构建过程中。众所周知，自从 2006 年至今深度神经网络的一个重要体现或要求便是训练数据量的规模要大(衡量标准可利用模型的参数个数与训练数据量的个数来比较)，由于以往训练数据集规模很小，加上计算性能很慢(硬件加速设备导致)，同时权值矩阵的初始化方式较为笨拙(容易出现梯度弥散现象)，以及使用了某种错误的非线性模型，导致深度神经网络在过去的表现并不好。经过十余年的积累，目前深度神经网络可简略地认为是大规模训练数据集，并行计算和规模化、灵巧的算法三者的结合。深度神经网络中引入稀疏正则或蕴含稀疏性可以认为是病态模型良态化的过程，如稀疏正则的核心是解决过拟合问题，稀疏权值连接(DropOut 策略)的本质是通过约减参数量间接增加训练数据，以及非线性激活函数中所隐含的稀疏性是为了增加“扭曲”程度，即不同类别的(线性不可分)输入随着层级的增加，隐层特征所对应的线性可分性逐渐增强。下面简要地分析深度神经网络在各阶段所出现的稀疏性及其优势。

备注：S-HMAX 为稀疏层次识别模型，SRC 为稀疏表示分类器，结构化稀疏模型，基于稀疏正则的设计有群稀疏、图稀疏、随机场稀疏等。

5.2.1　数据的稀疏性

数据的稀疏性包含三点：一是数据中所包含某种拓扑特性或目标相对数据本身呈现出非零元素较少的情形；二是数据在某种（线性或非线性）自适应或非自适应变换下对应的表示系数具有非零元素较少的状况；三是随着数据集规模的增加，呈现出某种统计或物理特性的数据占整个数据集的少数，例如分辨率特别好的样本或分辨率特别差的样本在整个数据集中呈较少的状态。目前，常用的稀疏性描述是基于第二点假设，并且作为一种有效的（稀疏性）正则约束，在优化目标函数关于解存在多样性的问题中给出合理的解释与逼近。而基于第一点，通常可作为一种有效的处理方式（如二值化处理，或者零化无关区域），例如输入到深度神经网络中的一幅图像，有效的目标占图像的比例较少，便可以将图像中除去目标的部分置为零；值得注意的是：利用视觉机制中的显著性检测方法。另外针对第三点，其核心问题是如何利用稀疏编码筛选出这些重要样本（或剔除少数样本）。从框架（Frame Analysis）分析角度，认为比较好的冗余框架应该是紧框架，进而对输入描述便可以得到较好的紧表示系数，也就是说框架上界和框架下界尽可能相等。但是通常获取到的字典，也就是框架，不是紧的，能否利用大量无类标样本将框架的上界与下界估计出来，然后利用输入信号的逼近表示的二范数比上表示系数的二范数，看这个比值是否在框架上界与下界的中间，来判断该样本对字典（框架或系统）的表示是否是 well-defined 的，进而实现对样本的有效筛选。

备注：本小节讲的框架，是数学分析中的一支理论，继傅里叶分析、时频分析和小波分析之后，框架分析被提出，它指带有冗余特性“基”的表示理论。

5.2.2　稀疏正则

众所周知，正则化的目的在于减少学习算法的泛化误差（亦称测试误差）以期提高测试识别率。目前，有许多正则化策略，常用的方式是对参数进行约束或限制，以及基于某种特定类型的先验知识进行约束与惩罚设计，注意这些惩罚和约束通过将模型求解参数良态化的过程来实现泛化性能的提升。基于如下的优化目标函数：

$$\min_{\theta} J(\theta) = \frac{1}{N}\sum_{n=1}^{N}\text{loss}(\boldsymbol{x}^{(n)}, y^{(n)}, \theta) + \lambda \cdot R(\theta) \tag{5.1}$$

其中的 $R(\theta)$ 为参数范数惩罚，例如常用的有 L_2 范数下的吉洪诺夫正则（Tikhonov Regularization），但它并没有蕴含稀疏特性。而使用 L_1 范数则通常可以诱导出稀疏特性，即

$$R(\theta) = \|\boldsymbol{W}\|_1 = \sum_i |W_i| \tag{5.2}$$

注意参数 θ 包括权值连接 $\boldsymbol{W}$ 与偏置 b，而正则约束往往只针对权值连接。除了在权值连接上引入稀疏正则外，还可以在某个隐层输出层引入稀疏性，例如对于如下的目标函数：

$$\min_{\vartheta} J(\vartheta) = \|\boldsymbol{x}-\boldsymbol{D}\cdot\vartheta\|_2^2+\lambda\cdot\|\vartheta\|_1 \tag{5.3}$$

注意这里的 $\boldsymbol{D}$ 为字典，数学中称其为框架，即有冗余的"基"；x 为输入，ϑ 为输出，其 L_1 范数的定义与式(5.2)对应。值得指出的是反卷积神经网络中的卷积稀疏编码可以认为是一种带有共享机制下的权值稀疏性约束策略。

备注：除了上述具有稀疏特性的 L_1 范数外，还可以引入群稀疏的策略，以及伪范数 $L_{1/2}$ 等，这里不再赘述。

5.2.3 稀疏连接

众所周知，卷积神经网络的特性包括局部连接，权值共享和变换不变等特性且都蕴含着稀疏性，首先针对局部连接，相比较全连接策略，它更符合外侧膝状体到初级视觉皮层上的稀疏响应特性；其次权值共享，进一步约束相似隐单元具有同样的激活特性，使得局部连接后的权值具有结构特性，实际应用中可进一步约减参数个数，间接增加数据量；最后，变换不变性是由池化方式诱导获取，也可认为是一种有效的"删减"参数的方式，即带有稀疏性的零化操作。下面介绍一种经典的自适应权值删减技巧 DropOut，即指在模型训练时随机让网络某些隐含层节点的权重不工作，不工作的那些节点可以暂时认为不是网络结构的一部分，但是它的权重需保留下来(注意只是暂时不更新)，因为下次样本输入时它可能又得工作了，见图 5.2。

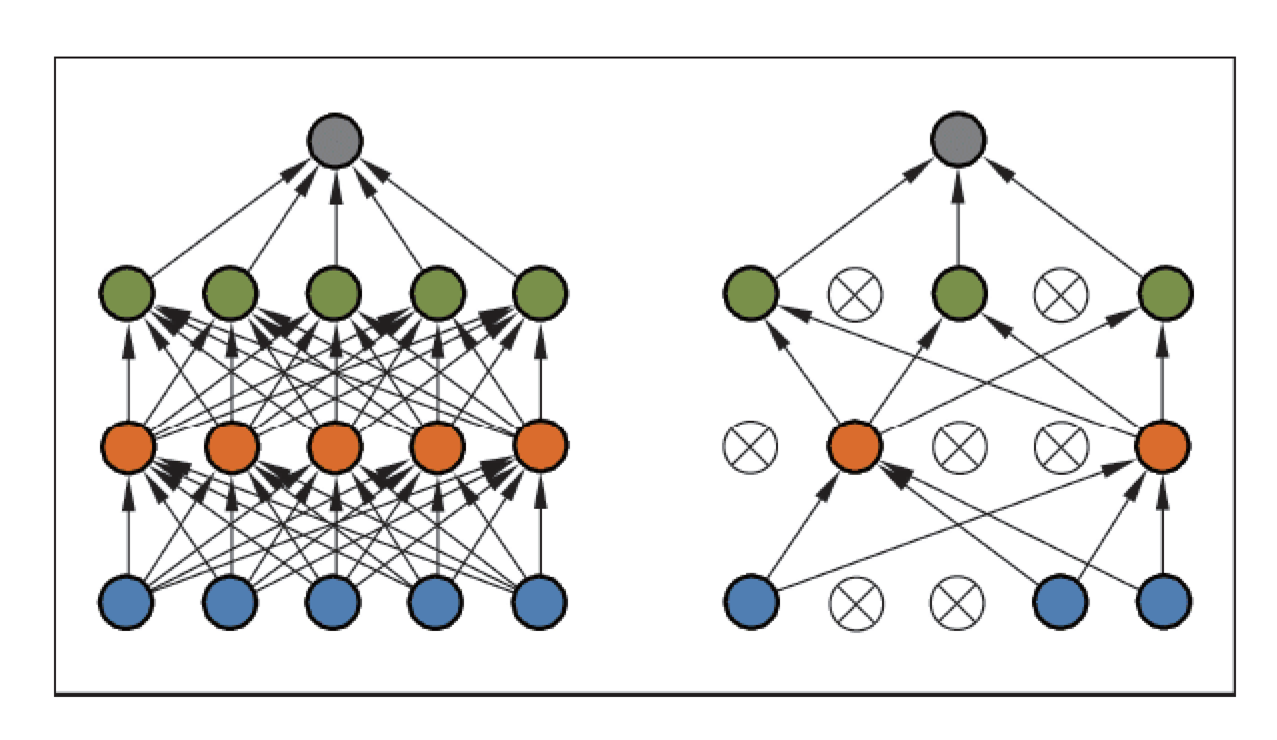


图 5.2 DropOut 网络连接

在图 5.2 的基础上，对第 l 层到第 $l+1$ 层上的第 i 个隐单元，在训练阶段，DropOut 具体的工作原理如图 5.3 所示。

其中图 5.3 中左边的网络结构为正常的连接，右边的为带有 DropOut 策略的连接，其数学物理解释如下。

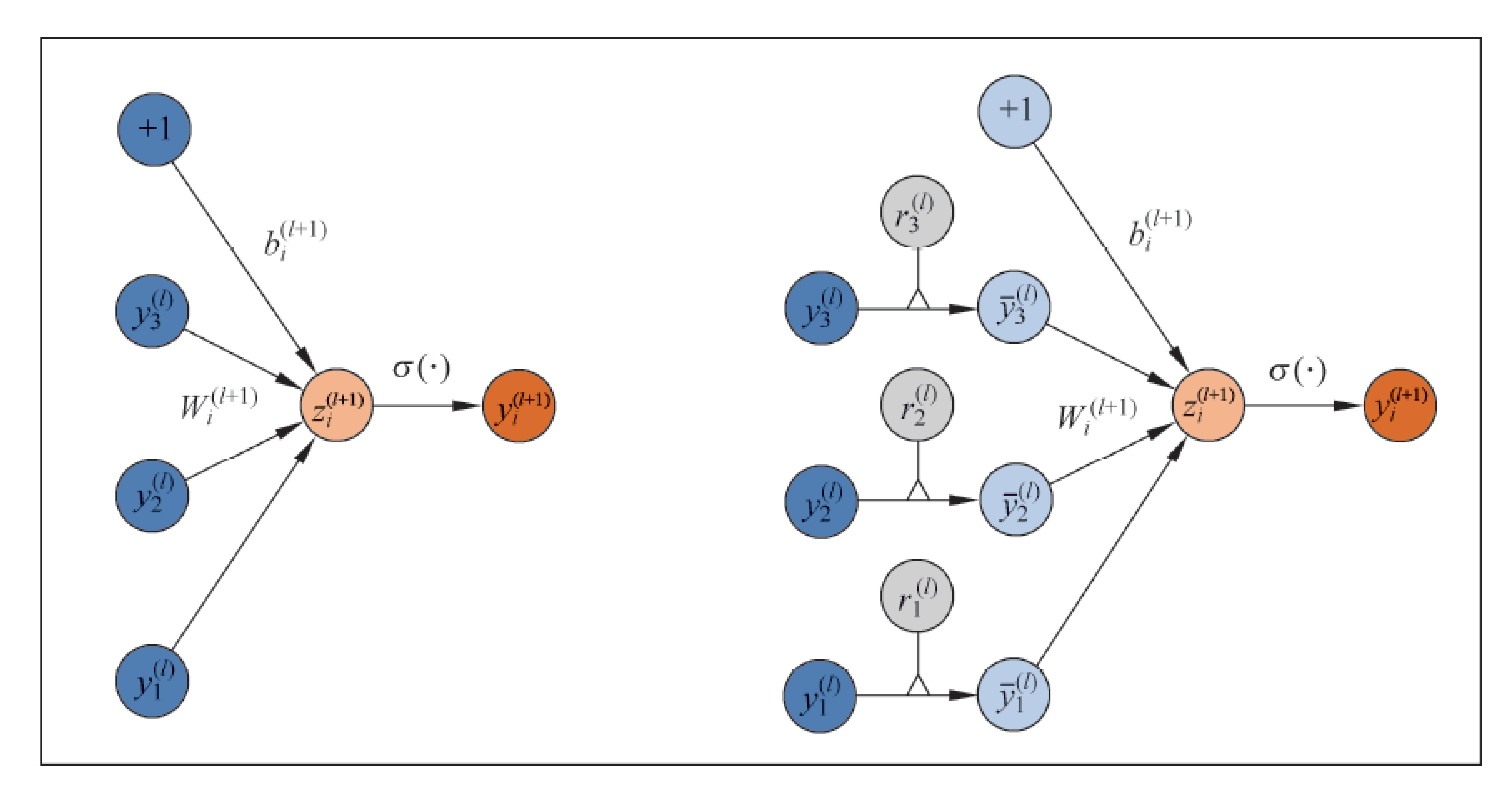


图 5.3　DropOut 的工作原理

1. 正常的连接

$$\begin{cases} z_i^{(l+1)} = \boldsymbol{W}_i^{(l+1)} \cdot y^{(l)} + b_i^{(l+1)} = \sum_{j=1}^{3} W_{i,j}^{(l+1)} \cdot y_j^{(l)} + b_i^{(l+1)} \\ y_i^{(l+1)} = \sigma(z_i^{(l+1)}) \end{cases} \tag{5.4}$$

注意其中权值连接为 $\boldsymbol{W}_i^{(l+1)} \in \mathbb{R}^3$，另外 $b_i^{(l+1)} \in \mathbb{R}$ 为偏置，$\sigma(\cdot)$为激活函数。

2. 带有 DropOut 策略的连接

$$\begin{cases} r_j^{(l)} \sim \text{Bernoulli}(p) \\ \tilde{\boldsymbol{y}}^{(l)} = \boldsymbol{r}^{(l)} \odot \boldsymbol{y}^{(l)} \in \mathbb{R}^3 \\ \boldsymbol{z}_i^{(l+1)} = \boldsymbol{W}_i^{(l+1)} \cdot \tilde{\boldsymbol{y}}^{(l)} + b_i^{(l+1)} = \sum_{j=1}^{3} W_{i,j}^{(l+1)} \cdot \tilde{y}_j^{(l)} + b_i^{(l+1)} \\ y_i^{(l+1)} = \sigma(\boldsymbol{z}_i^{(l+1)}) \end{cases} \tag{5.5}$$

其中符号⊙为对应元素相乘，另外，伯努利(Bernoulli)分布是一种离散分布，有两种可能的结果，其中 1 表示成功，0 表示失败，注意符号 p 表示概率值，即 $r_j^{(l)}$ $(j=1,2,3)$是以概率 p 成功响应的。对比式(5.5)和式(5.4)可知，从输入 $y^{(l)}$ 到 $\tilde{y}^{(l)}$，导致第 l 层上部分节点不响应，注意由于每个节点是独立同分布下的响应或不响应，所以处理完后响应节点的个数为：

$$\tilde{n}_l = n_l \cdot p \tag{5.6}$$

其中 p 为相应概率，即 DropOut 率；n_l 为隐层节点的个数，$\tilde{n}_l$ 为随机概率处理完后的第 l 层上的响应节点的个数。应用中，经过交叉验证，隐含节点 Dropout 率等于 0.5 的时候效果最好，主要原因是此时 Dropout 随机生成的网络结构最多。

另一种稀疏连接可以通过约减参数的方式来实现，通常有两个思路：一是直接将较小

的权值连接置为零(但有风险,因为随着层级的上升,较小的权值将会使得输入累积较为大的输出);二是通过矩阵分解来实现。下面简要介绍基于矩阵分解的参数约减,假设输入 $\boldsymbol{x}\in\mathbb{R}^m$ 与输出 $\boldsymbol{y}\in\mathbb{R}^n$ 之间的关系为:

$$\begin{cases}\boldsymbol{y}=\sigma(\boldsymbol{W}\cdot\boldsymbol{x}+b)\\ \boldsymbol{W}\in\mathbb{R}^{n\times m}\end{cases}\tag{5.7}$$

其中 $\boldsymbol{W}$ 为权值连接,$b\in\mathbb{R}$ 为偏置。进一步,对于权值连接 $\boldsymbol{W}$ 通过奇异值矩阵分解得到:

$$\boldsymbol{W}=\boldsymbol{U}\cdot\boldsymbol{\Sigma}\cdot\boldsymbol{V}^{\mathrm{T}}\tag{5.8}$$

这里假设 $\mathrm{rank}(\boldsymbol{W})=r$,则有 $\boldsymbol{U}\in\mathbb{R}^{n\times r}$,$\boldsymbol{\Sigma}\in\mathbb{R}^{r\times r}$ 和 $\boldsymbol{V}\in\mathbb{R}^{m\times r}$;通过组合策略得到权值连接的表示为:

$$\begin{cases}\boldsymbol{W}=\boldsymbol{W}_2\cdot\boldsymbol{W}_1\\ \boldsymbol{W}_2\in\mathbb{R}^{n\times r}\\ \boldsymbol{W}_1\in\mathbb{R}^{r\times m}\end{cases}\tag{5.9}$$

注意当 $\boldsymbol{W}_1=\boldsymbol{U}\cdot\boldsymbol{\Sigma}$ 时,则 $\boldsymbol{W}_2=\boldsymbol{V}^{\mathrm{T}}$;抑或当 $\boldsymbol{W}_1=\boldsymbol{U}$ 时,则 $\boldsymbol{W}_2=\boldsymbol{\Sigma}\cdot\boldsymbol{V}^{\mathrm{T}}$;模型相对应的式(5.7)则变为:

$$\begin{cases}\boldsymbol{z}=\boldsymbol{W}_1\cdot\boldsymbol{x}\\ \boldsymbol{y}=\sigma(\boldsymbol{W}_2\cdot\boldsymbol{z}+b)\end{cases}\tag{5.10}$$

需要注意的是:网络模型相对应式(5.7)中的权值连接 $\boldsymbol{W}$ 和式(5.10)中的权值连接($\boldsymbol{W}_1$,$\boldsymbol{W}_2$),其参数量由 $n\cdot m$ 变为 $r\cdot(n+m)$。注意该规则有效的前提是权值连接 $\boldsymbol{W}$ 是低秩矩阵,即 $\mathrm{rank}(\boldsymbol{W})<\min(n,m)$。

备注:由于在实际大多数情形下,权值矩阵 $\boldsymbol{W}$ 是满秩的,因此通常取 $\boldsymbol{\Sigma}$ 的较大的 k 个奇异值并将其他奇异值置零,来实现对 $\boldsymbol{W}$ 的逼近。

5.2.4 稀疏分类器设计

常见的稀疏分类器设计是基于表示学习的,如稀疏表示分类器,其核心步骤包括:首先,字典构造:

$$\boldsymbol{D}=[\boldsymbol{D}_1,\boldsymbol{D}_2,\cdots,\boldsymbol{D}_K]\tag{5.11}$$

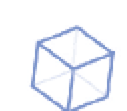
其中 K 为类别个数,$\boldsymbol{D}_k(k=1,2,\cdots,K)$为第 k 类样本或数据集构造(直接将样本堆叠形成)或学习(通过 K-SVD 算法)的字典;其次,对于样本 $\boldsymbol{x}$ 进行如下的表示学习:

$$\begin{aligned}&\min_{\boldsymbol{\alpha}}\frac{1}{2}\cdot\|\boldsymbol{x}-\boldsymbol{D}\cdot\boldsymbol{\alpha}\|_2^2+\lambda\cdot\|\boldsymbol{\alpha}\|_1\\ &=\frac{1}{2}\cdot\left\|\boldsymbol{x}-\sum_{k=1}^{K}\boldsymbol{D}_k\cdot\boldsymbol{\alpha}_k\right\|_2^2+\lambda\cdot\|\boldsymbol{\alpha}\|_1\end{aligned}\tag{5.12}$$

注意这里的表示系数:

$$\boldsymbol{\alpha}=[\boldsymbol{\alpha}_1,\boldsymbol{\alpha}_2,\cdots,\boldsymbol{\alpha}_K]^{\mathrm{T}}\tag{5.13}$$

其中基于假设,若样本 $\boldsymbol{x}$ 属于第 k 类,则表示系数主要集中在 $\boldsymbol{\alpha}_k$,而其他表示系数 $\boldsymbol{\alpha}_j(j\neq k)$期



望为零，需要注意的是，这里的$\boldsymbol{\alpha}_k$是向量，而不是标量。最后类标的判定通过如下的公式实现：

$$\text{label}(\boldsymbol{x})=\arg\min_{1\leqslant k\leqslant K}\{\|\boldsymbol{x}-\boldsymbol{D}_k\cdot\boldsymbol{\alpha}_k\|_2^2\} \tag{5.14}$$

另一种稀疏分类器的设计则是基于改进的Softmax分类器来实现的，其动机是改进Softmax输出处处不为零以期获得输出大多数为零，并记此为Sparsemax分类器，具体的数学物理描述如下。

1. Softmax分类器

$$\begin{cases}\boldsymbol{y}=\text{Softmax}(\boldsymbol{x},\theta)=[y_1,y_2,\cdots,y_K]^{\mathrm{T}}\\ y_k=P(\text{label}(\boldsymbol{x})=k\mid\boldsymbol{x},\theta_k)=\dfrac{1}{Z}\cdot\mathrm{e}^{(\boldsymbol{x}\cdot\theta_k)}\end{cases} \tag{5.15}$$

其中K为类别个数，参数$\theta=[\theta_1,\theta_2,\cdots,\theta_K]$，$Z$为归一化因子，即

$$Z=\sum_{j=1}^{K}\mathrm{e}^{(\boldsymbol{x}\cdot\theta_j)} \tag{5.16}$$

待优化的参数为θ。

2. Sparsemax分类器

$$\begin{cases}\boldsymbol{y}=\text{Sparsemax}(\boldsymbol{x},\vartheta)=\arg\min\limits_{\boldsymbol{p}\in\Delta^{k-1}}\|\boldsymbol{p}-(\boldsymbol{W}\cdot\boldsymbol{x}+b)\|_2^2\\ \vartheta=(\boldsymbol{W},b)\end{cases} \tag{5.17}$$

其中这里的“Δ^{k-1}”为单形，即满足：

$$\Delta^{k-1}=\left\{\boldsymbol{p}\in\mathbb{R}^K:\sum_{i=1}^{K}p(i)=1,p(i)\geqslant 0\right\} \tag{5.18}$$

如何优化求解p？对于Softmax分类器而言，已知参数θ和输入$\boldsymbol{x}$，则可以通过式(5.15)求出输出$\boldsymbol{y}$。而对于Sparsemax分类器，若知参数ϑ和输入$\boldsymbol{x}$，如何对式(5.17)进行优化呢？先简记符号：

$$\boldsymbol{z}=\boldsymbol{W}\cdot\boldsymbol{x}+b \tag{5.19}$$

则优化目标变为：

$$\arg\min_{\boldsymbol{p}\in\Delta^{k-1}}\|\boldsymbol{p}-\boldsymbol{z}\|^2 \tag{5.20}$$

给定$\boldsymbol{z}$，关于$\boldsymbol{p}\in\Delta^{k-1}$有如下形式的解：

$$p_k=[z_k-\tau(\boldsymbol{z})]_+ \tag{5.21}$$

这里的p_k为将输入$\boldsymbol{x}$分到第k类的概率，其中$k=1,2,\cdots,K$和$[t]_+=\max(0,t)$。

另外$\tau:\mathbb{R}^K\to\mathbb{R}$满足以下式子：

$$\sum_k[z_k-\tau(\boldsymbol{z})]_+=1 \tag{5.22}$$



进一步，记 z 中的坐标排序为：

$$z_1 \geqslant z_2 \geqslant \cdots \geqslant z_K \tag{5.23}$$

并定义：

$$k(z) = \max\left\{k \in [1,2,\cdots,K]: 1 + k \cdot z_k \geqslant \sum_{j \leqslant k} z_j\right\} \tag{5.24}$$

则 $\tau(\cdot)$可以被如下表示：

$$\tau(z) = \frac{\left(\sum_{j \leqslant k(z)} z_j\right) - 1}{k(z)} \tag{5.25}$$

5.2.5　深度学习中关于稀疏的技巧与策略

众所周知，深度学习是一类借鉴生物的多层神经网络处理模式所发展起来的智能处理技术，稀疏性可以大幅度削减深度神经网络中权值连接数量，因此被广泛采用。目前，对于深度卷积神经网络，便可以认为是深度前馈（全连接）神经网络的稀疏化；另外，稀疏深度网络模型的设计包括以下三条准则。

（1）第一条准则，层级间模块化，逐层堆栈。

依据自编码网络进行逐层初始化，例如常用的有稀疏自编码器，其中关于隐结点输出的稀疏正则性约束包括 KL 散度和 L1 范数或伪范数，对应着的稀疏深度网络模型称为（稀疏）深度堆栈网络；另外还有稀疏受限玻尔兹曼机所对应的稀疏深度置信网络和卷积稀疏编码所对应的（稀疏）反卷积神经网络等。

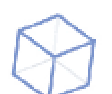
（2）第二条准则，逐阶段模块化。

与层级间模块化不同，针对特定的任务，例如分类，利用生成式对抗网络（包括两个子网络，即生成模型和判别模型）在无监督学习方式下获取非合作状态下的零和博弈解，提取其判别网络中的特征学习部分（去掉后面的真伪二值分类器设计），结合分类器设计（如 Softmax 分类器），再利用监督学习的方式进行整个网络（由提取的特征学习部分和分类器设计部分组合而成）的精调。其中稀疏化可以内蕴在特征学习部分和分类器中。

（3）第三条准则，多通路网络设计。

多分辨特性可以认为是输入在不同尺度或不同频带上的响应，相比较单尺度上对输入的（稠密性）表征，多分辨特性通过多通路或多通道来散化对输入的表征，使其在每一个尺度或频带上呈现稀疏性。另外，根据深度神经网络的设计准则，塔式、对称和多通路可以削弱“深度”对输入与输出之间的非线性刻画，即极深神经网络（例如深度残差网络、深度分形网络等）可由多通路、带有融合特性的深度神经网络来逼近。

在以上三条准则的基础上，常使用的稀疏性策略包括 Dropout（目的是通过随机化权值连接实现参数的有效约减，间接提升训练数据量，以实现网络泛化性能的提升，有助于防止



过拟合现象），DropConnect，DropNeuron 等；另外，网络中激活函数的有效选择将有助于通过内蕴稀疏性来提升网络的泛化性能和计算开销，以及缓解反向传播时所带来的“梯度弥散”的现象，常用的激活函数见图 5.4。

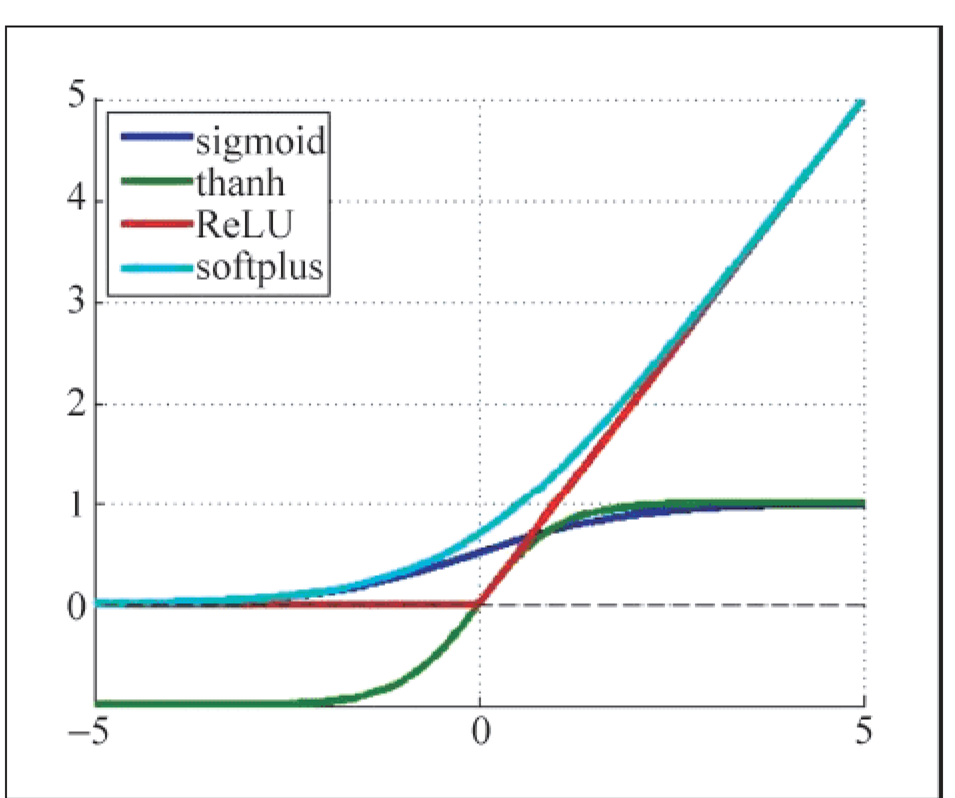


图 5.4 常用的激活函数

目前，深度网络设计中最为常用的激活函数是修正线性单元 ReLU 及其改进版 RePLU，Maxout（本质上 ReLU 是 Maxout 的一种特例，操作见图 5.5）等。值得注意的是：深度学习的基础是数据，由于数据本身存在着差异性，对深度网络模型的影响也不一样；能否通过对数据的“分级处理”，如常见的基于无监督方式的数据聚类，通过划定与聚类中心的亲疏来实现样本的分级处理，如“优良中差”子数据集；进一步，对每级样本分别来学习深度神经网络，以期探索数据的差异性对深度卷积神经网络的影响。换言之，数据的分级处理体现着输入与输出之间映射的差异性，犹如大脑的多分辨特性，对信息结构完整或分辨率高的输入识别精度高，相反，对结构缺失或分辨率较低的输入识别精度低；若将这种多分辨特性与深度卷积神经网络相结合，形成多分辨深度卷积神经网络，实现对样本的筛选并改善基于差异性数据集学习到的深度卷积神经网络的性能。

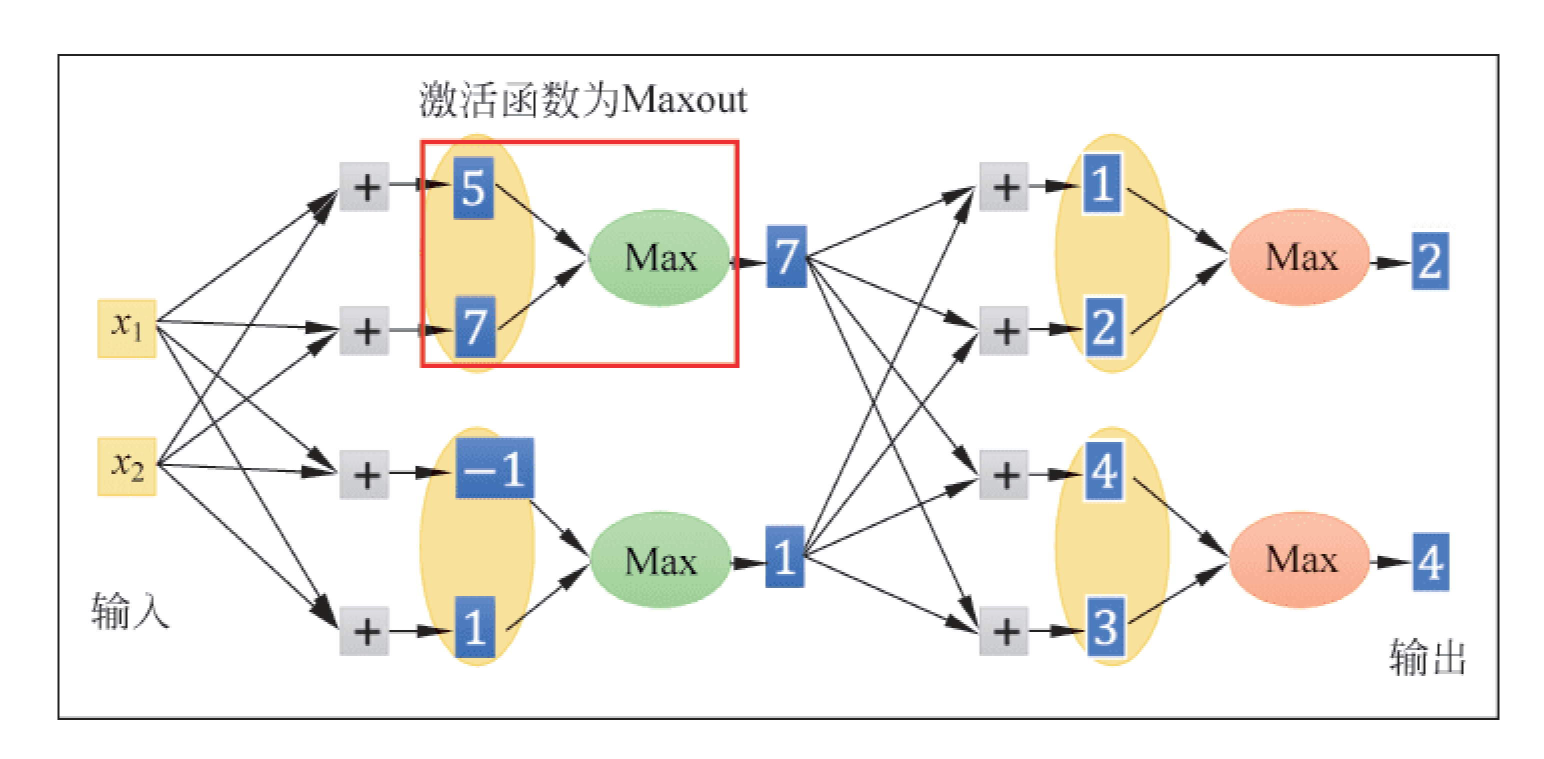


图 5.5 激活函数使用 Maxout

备注：关于激活函数 ReLU 中蕴含着的稀疏性请参考链接 http://www.cnblogs.com/yymn/p/5616709.html。

注意：Maxout 与 ReLU 的唯一区别是，前者是对若干个“隐隐层”单元的值执行最大化操作，而后者是对隐层上每一个单元执行与 0 相比较的最大化操作。

5.3 网络模型的性能分析

5.3.1 稀疏性对深度学习的影响

通常，原始数据中缠绕着高度密集的特征，稠密分布内蕴稀疏表达往往比局部少数点携载的特征成倍地有效，当然，在网络的设计过程中，过分地强调稀疏性处理，会减少模型的有效容量，即特征屏蔽太多，导致模型无法学到有效的特征；研究发现，理想的稀疏性比率保持在70%～85%(量化的指标说明请参考备注中的参考文献)，超过85%的深度网络模型的网络容量就成了问题，导致泛化性能锐减错误率极高。总之，模型稀疏化有诸多优点，但是过度的(显式)稀疏性通常也会导致模型的稳定性变差，从而泛化性能降低。

5.3.2 对比试验及结果分析

本节简要地给出稀疏深度堆栈网络的几组实验说明及结果分析。首先，网络的结构见图5.6，网络优化分为预训练和精调两个阶段，超参数中关于激活函数和特征学习后分类器的设计作为对比点。

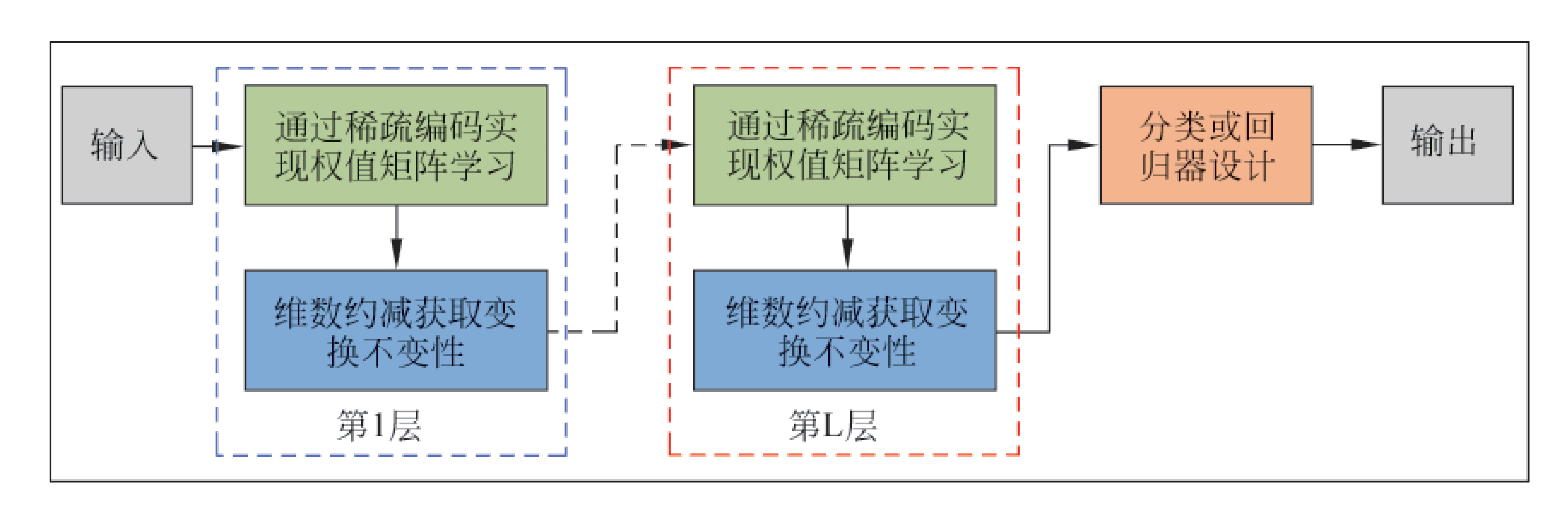


图5.6 稀疏深度堆栈网络

1. 激活函数所蕴含着的稀疏性(分类器固定为Softmax)(表5.2)

表5.2 不同激活函数下稀疏深度堆栈网络的测试误差

激活函数	Mnist	Cifar10	Mstar	ImageNet
Sigmoid	3.28%	48.12%	12.18%	58.65%
Tanh	3.16%	51.04%	11.97%	56.92%
Softplus	2.75%	43.76%	10.34%	54.21%
ReLU	2.93%	46.23%	10.93%	55.63%
Maxout	2.64%	44.74%	9.73%	52.17%

注意网络的隐层个数设计为3层,预训练阶段,权值初始化的方式采用稀疏编码策略,即将学到的字典进行转置得到相应的滤波器,通过最大池化的方式进行维数约减。

2. 稀疏分类器设计 VS 分类器设计(激活函数使用 ReLU)(表 5.3)

表 5.3　不同分类器下稀疏深度堆栈网络的测试误差

分　类　器	Mnist	Cifar10	Mstar	ImageNet
SVM	2.68%	43.47%	9.10%	54.96%
SMM	2.57%	42.71%	8.92%	54.05%
Softmax	2.93%	46.23%	10.93%	55.63%
Sparsemax	2.52%	44.38%	7.52%	55.15%

注意这里的 SMM 为支撑矩阵机,详尽参考第1章机器学习小结中支撑向量机第二种改进的方案。

结果分析,从实验中可以看出 ReLU 激活函数隐含对特征"非负稀疏性"的要求;而(最大)池化操作隐含对特征"强稀疏性"(特征选择)的要求;以及参数层偏置隐含对特征"稀疏度"的调节。另外,深度神经网络是关于自动学习要建模的数据的潜在(隐含)分布的多层(复杂)表达的算法。换言之,深度学习算法自动地提取分类需要的高层抽象特征,而适当地引入显式或隐式稀疏规则将有助于克服过拟合现象的发生,同时提升网络的泛化性能。

参考文献

[5.1] Hu X, Zhang J, Li J, et al. Sparsity-regularized HMAX for visual recognition[J]. Plos One, 2014, 9(1): e81813.

[5.2] Wright J, Yang A Y, Ganesh A, et al. Robust Face Recognition via Sparse Representation[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2009, 31(2): 210-227.

[5.3] Martins A F T, Astudillo R F. From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification[J]. 2016.

[5.4] Glorot X, Bordes A, Bengio Y. Deep Sparse Rectifier Neural Networks[J]. Journal of Machine Learning Research, 2011, 15.

[5.5] Toth L. Phone recognition with deep sparse rectifier neural networks[C]. IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013: 6985-6989.

[5.6] Pironkov G, Dupont S, Dutoit T. Investigating sparse deep neural networks for speech recognition [C]. Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding. IEEE, 2015: 124-129.

[5.7] Scardapane S, Comminiello D, Hussain A, et al. Group sparse regularization for deep neural networks[J]. Neurocomputing, 2016.

[5.8] Zheng H, Chen M, Liu W, et al. Improving deep neural networks by using sparse dropout strategy [C]. IEEE China Summit & International Conference on Signal and Information Processing. IEEE,

2014：21-26.

[5.9] Han S，Pool J，Narang S，et al. DSD：Regularizing Deep Neural Networks with Dense-Sparse-Dense Training Flow[J]. 2016.

[5.10] Graham B. Spatially-sparse convolutional neural networks[J]. Computer Science，2014，34(6)：864-867.

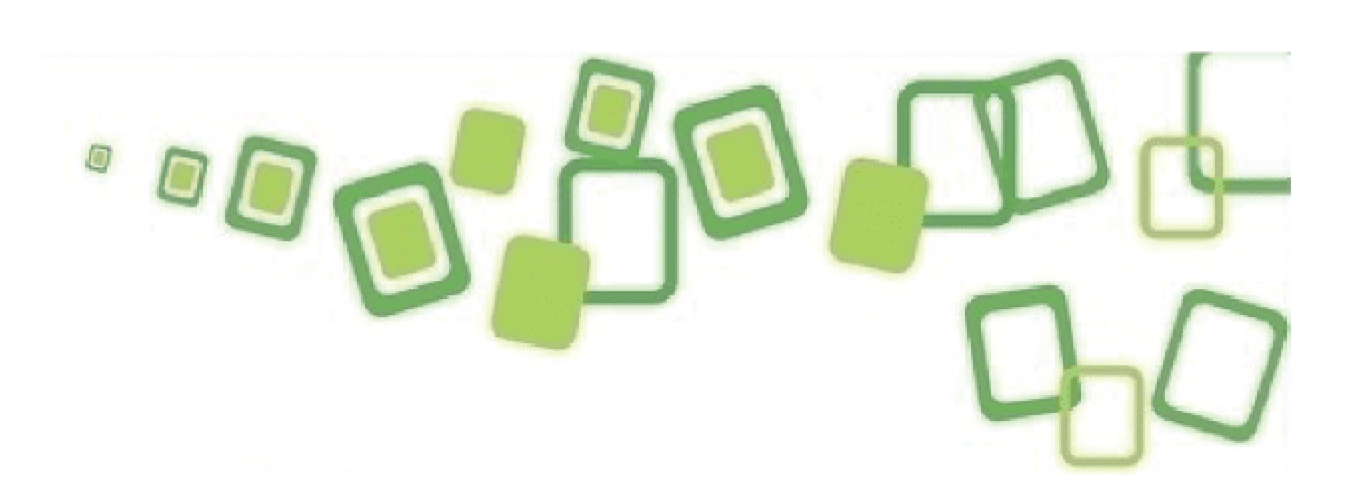
[5.11] Liu B，Wang M，Foroosh H，et al. Sparse Convolutional Neural Networks[C]. Computer Vision and Pattern Recognition. IEEE，2015：806-814.

[5.12] Gripon V，Berrou C. Sparse neural networks with large learning diversity[J]. IEEE Transactions on Neural Networks，2011，22(7)：1087-1096.

[5.13] Balavoine A，Romberg J，Rozell C J. Convergence and Rate Analysis of Neural Networks for Sparse Approximation[J]. IEEE Transactions on Neural Networks & Learning Systems，2011，23(9)：1377-1389.

[5.14] Pernice V，Rotter S. Reconstruction of sparse connectivity in neural networks from spike train covariances [J]. Journal of Statistical Mechanics Theory & Experiment，2013，2013(2013)：P03008.

第6章

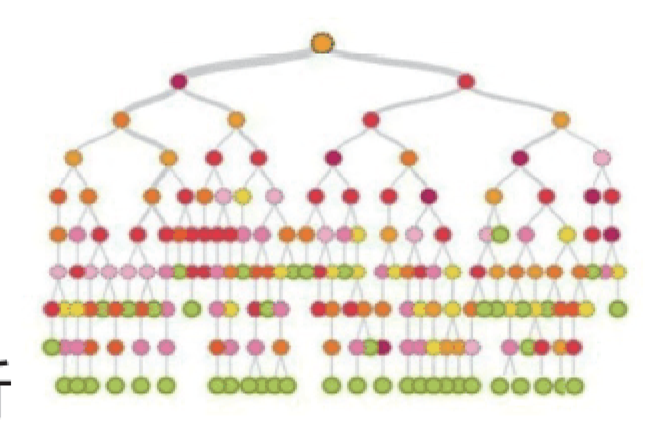


深度融合网络

CHAPTER 6

深度学习融合机器学习方法

- 深度SVM网络——SVM
- 深度PCA网络——PCA
- 深度层级识别网络S-HMAX——稀疏编码、表示
- 深度ICA网络——ICA
- 深度ADMM网络——ADMM算法
- 深度极限学习机——极限学习机ELM
- 深度森林——决策树、随机森林
- 深度小波网络——小波分析
- 深度多尺度几何网络——多尺度几何分析



6.1　深度 SVM 网络

在本节中，将深度学习中的“深度”含义与机器学习中经典的支撑向量机(Support Vector Machine，SVM)算法相结合，形成深度 SVM 网络，或也称深度神经支撑向量机。下面从网络模型形成的动机、拓扑结构(数学刻画)和实用训练技巧等三个方面详述该网络。

6.1.1　从神经网络到 SVM

首先结合图 6.1，从数学角度给出经典的三层前馈神经网络与支撑向量机 SVM 的区别与联系.

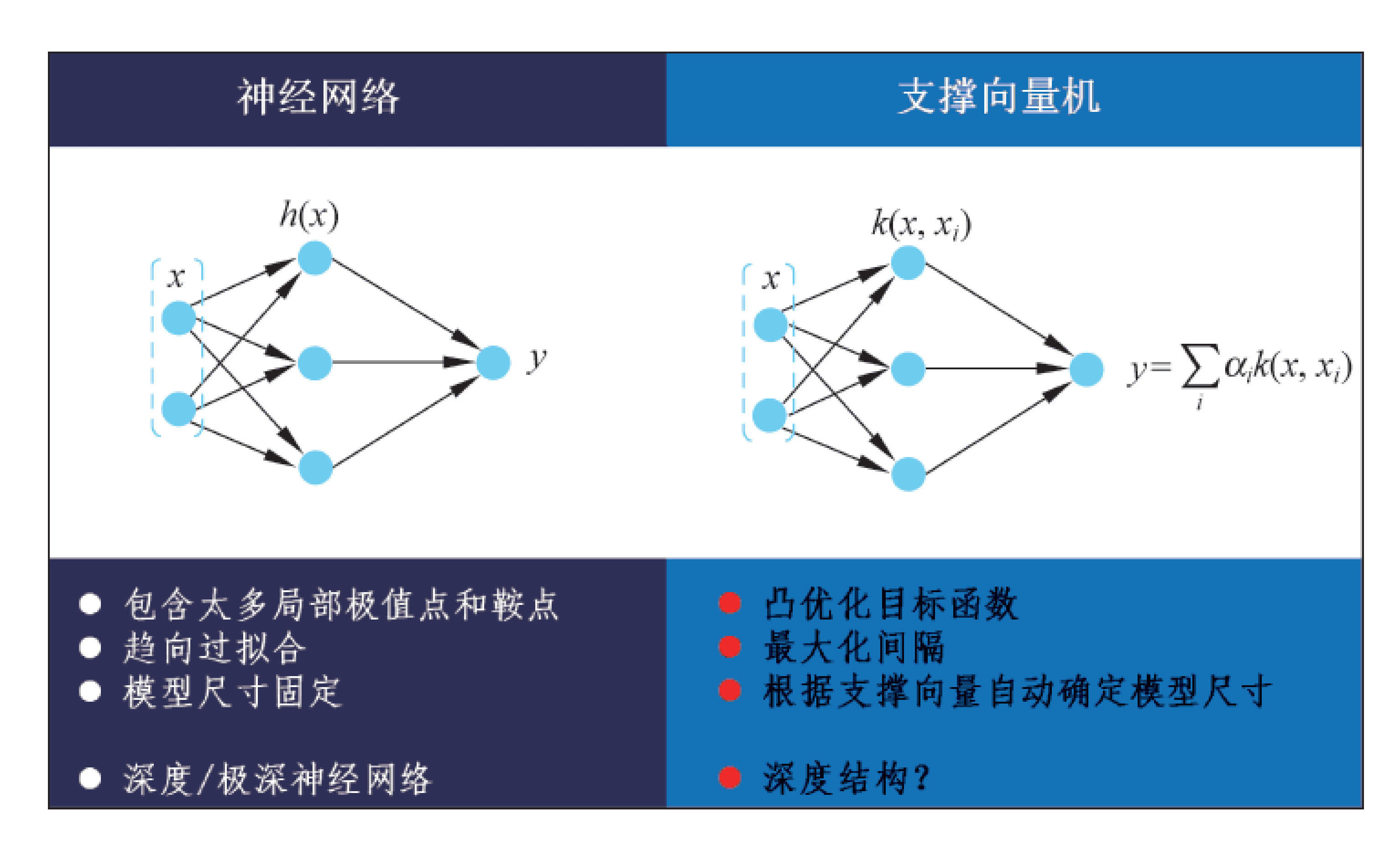


图 6.1　神经网络与支撑向量机 SVM 的对比

1. 神经网络

经典的三层前馈神经网络模型(输入与输出之间的关系)为：

$$\begin{cases} h(\boldsymbol{x}) = \sigma_1(\boldsymbol{W} \cdot \boldsymbol{x} + b) \\ \boldsymbol{y} = \sigma_2(\boldsymbol{\omega} \cdot h(\boldsymbol{x}) + \beta) \end{cases} \tag{6.1}$$

其中 $\sigma_1(\cdot)$和 $\sigma_2(\cdot)$为激活函数(非线性函数)，$h(\boldsymbol{x})$为输入 $\boldsymbol{x}$(学到)的隐层特征。通常，基于该模型所构造的优化目标函数，其中待优化参数 $\theta=(\boldsymbol{W},b;\boldsymbol{\omega},\beta)$的可行域包含太多的局部极值点和鞍点；另外，该模型的训练性能和泛化性能严重依赖数据量且容易出现过拟合现象，并且网络的设计(层数、隐单元个数和非线性函数等超参数的设置以及训练阶段参数初始化、学习率等的给定)随着人为给定而固定，虽然可以利用格式搜索的方法确定超参数，但计算代价太大。



2. SVM

经典的线性 SVM 网络模型为：

$$\boldsymbol{y} = \boldsymbol{W} \cdot \boldsymbol{x} + b \tag{6.2}$$

若训练数据集为

$$\{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N} \tag{6.3}$$

根据第 1 章中 SVM 的介绍，利用其对偶问题关于变量偏导数求导，得到：

$$\boldsymbol{W} = \sum_{n=1}^{N} a^{(n)} \cdot \boldsymbol{y}^{(n)} \cdot \boldsymbol{x}^{(n)} \tag{6.4}$$

式(6.2)可进一步改写为：

$$\boldsymbol{y} = \boldsymbol{W} \cdot \varphi(\boldsymbol{x}) + b \tag{6.5}$$

其中 $\varphi(\boldsymbol{x})$为 $\boldsymbol{x}$ 的特征(通过特征学习获取)，则：

$$\boldsymbol{W} = \sum_{n=1}^{N} a^{(n)} \cdot \boldsymbol{y}^{(n)} \cdot \varphi(\boldsymbol{x}^{(n)}) \tag{6.6}$$

进而有：

$$\boldsymbol{y} = \sum_{n=1}^{N} a^{(n)} \cdot \boldsymbol{y}^{(n)} \cdot \langle \varphi(\boldsymbol{x}^{(n)}), \varphi(\boldsymbol{x}) \rangle + b \tag{6.7}$$

根据核函数的定义，可得到：

$$\begin{cases} \boldsymbol{y} = \sum_{n=1}^{N} a^{(n)} \cdot \boldsymbol{y}^{(n)} \cdot \kappa(\boldsymbol{x}^{(n)}, \boldsymbol{x}) + b \\ \kappa(\boldsymbol{x}^{(n)}, \boldsymbol{x}) = \langle \varphi(\boldsymbol{x}^{(n)}), \varphi(\boldsymbol{x}) \rangle \end{cases} \tag{6.8}$$

为了方便，且由于输出 $\boldsymbol{y}^{(n)}$ 已知，对应的模型进一步简记为：

$$\boldsymbol{y} = \sum_{n=1}^{N} \alpha_n \cdot \kappa(\boldsymbol{x}, \boldsymbol{x}^{(n)}) \tag{6.9}$$

即图 6.1 中的显示。众所周知，SVM 所得到的优化目标函数为凸优化问题且模型结构中内蕴：根据支撑向量(训练数据集的不同，对应着的支撑向量的个数也不一样)自动确定模型的尺寸。

那么能否根据神经网络到深度神经网络的思路，将 SVM 也对应着形成深度网络模型，即深度 SVM 网络？回答是肯定的，M. A. Wiering 等人于 2013 年提出了深度 SVM 网络。

6.1.2　网络模型的结构

首先，简单地给出深度 SVM 网络的结构(见图 6.2)，核心在于非线性单元的设计，下面从数据、模型、优化目标函数和求解四个方面详述并理解该网络，对应的任务为回归逼近。

1. 训练数据集

$$\{\boldsymbol{x}^{(n)} \in \mathbb{R}^{m}, \boldsymbol{y}^{(n)} \in \mathbb{R}^{s}\}_{n=1}^{N} \tag{6.10}$$

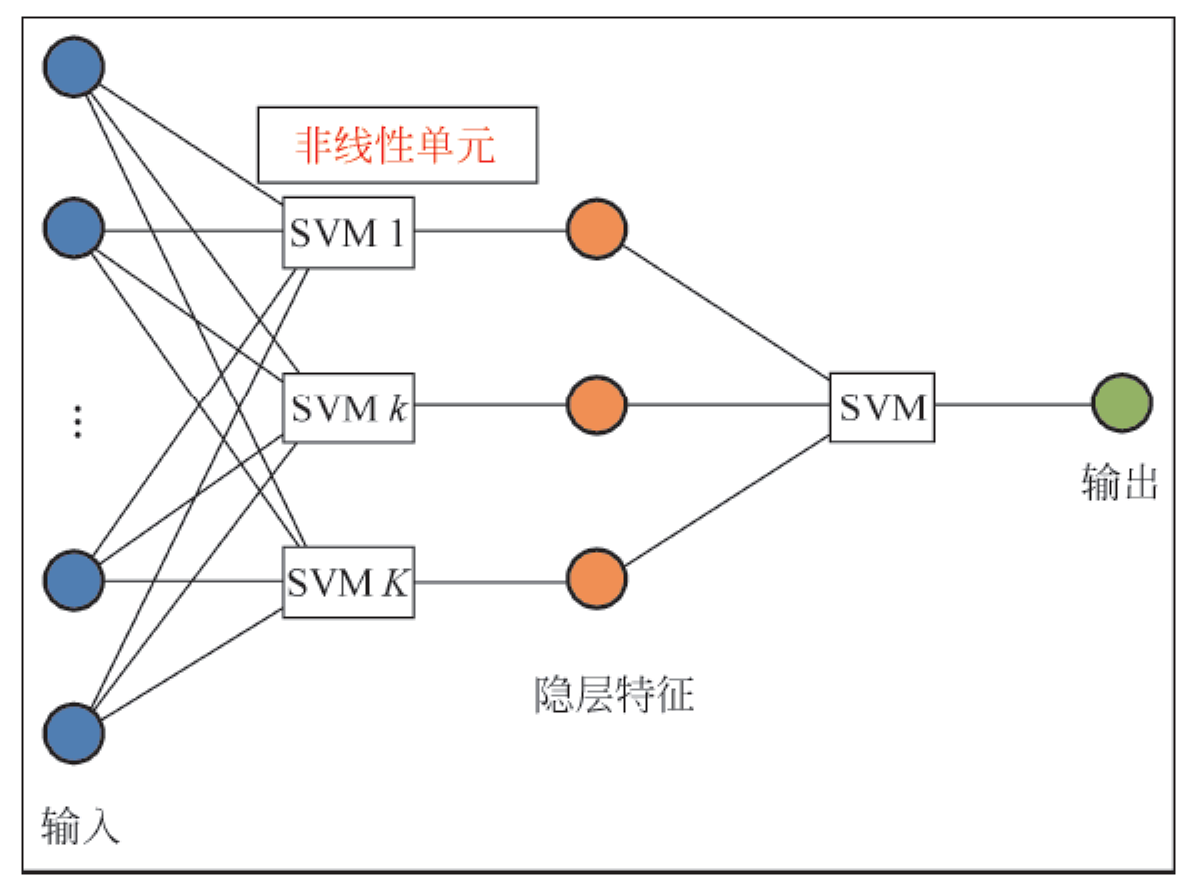


图 6.2 深度 SVM 网络结构

2. 模型

设模型输入为 $\boldsymbol{x}$，输出为 $\boldsymbol{y}$，其中的关系为：

$$\begin{cases} \boldsymbol{h} = (h_1, h_2, \cdots, h_K) \\ h_k = \sum_{n=1}^{N} \alpha_k^{(n)} \kappa(\boldsymbol{x}^{(n)}, \boldsymbol{x}) + b_k \end{cases} \tag{6.11}$$

其中 $\kappa(\cdot)$为核函数，根据 SVM 隐层每个节点的输出 $h_k(k=1,2,\cdots,K)$，且 $\alpha_k^{(n)}$ 和 b_k 为第 k 个待学习(通过 SVM 优化目标函数)的参数与偏置；值得注意的是：h_k并不是一维的(通过式(6.7)理解)，根据需要，可以随意定义。接下来将隐层特征 $\boldsymbol{h}$ 作为下一个非线性单元(SVM 单元)的输入，即有：

$$\begin{cases} \boldsymbol{y} = \sum_{n=1}^{N} \beta^{(n)} \cdot \kappa(\boldsymbol{h}^{(n)}, \boldsymbol{h}) + c \\ \boldsymbol{h}^{(n)} = [h_1^{(n)}, h_2^{(n)}, \cdots, h_K^{(n)}] \\ \boldsymbol{h} = [h_1, h_2, \cdots, h_K] \end{cases} \tag{6.12}$$

注意 $\boldsymbol{h}^{(n)}$ 为输入 $\boldsymbol{x}^{(n)}$ 的隐层特征，$\boldsymbol{h}$ 为输入 $\boldsymbol{x}$ 的隐层特征，其中待学习的参数为 $\beta^{(n)}$ 和 c。

3. 优化目标函数

优化目标函数为

$$\min_{\theta} J(\theta) = \frac{1}{N} \sum_{n=1}^{N} \| \hat{\boldsymbol{y}}^{(n)} - \boldsymbol{y}^{(n)} \|_2^2 + \lambda \cdot R(\theta) \tag{6.13}$$

其中参数为：

$$\begin{cases} \theta = (\alpha, b; \beta, c) \\ \alpha = \{\alpha_k^{(n)}\}_{n,k=1}^{N,K}; b = \{b_k\}_{k=1}^{K} \end{cases}$$

符号 $R(\theta)$为正则项(例如可以加入稀疏正则项等)。

4. 求解

采用梯度下降的方式实现参数的优化学习,其核心便是误差传播项的偏导数求解,由于图 6.2 中的网络结构仅包含一个隐层,所以误差传播项为:

$$\delta = \frac{\partial J(\theta)}{\partial \boldsymbol{h}} = \left(\frac{\partial J(\theta)}{\partial h_1}, \frac{\partial J(\theta)}{\partial h_2}, \cdots, \frac{\partial J(\theta)}{\partial h_K}\right) \tag{6.14}$$

然后利用链式法则,进行逐层参数的更新(与之前深度前馈神经网络的更新策略类似,这里不再赘述)。

通过以上四个方面的分析,可以看到模型的深度可以通过式(6.11)实现扩展,直至形成深度 SVM 模型,严格意义上,图 6.2 中的网络结构(仅包含一个隐层,或两个层级下的多 SVM 模式的组合)不应称为深度 SVM 模型。

6.1.3 训练技巧

针对中小规模的十种不同的数据集,利用 SVM 和深度 SVM 分别进行回归任务逼近,研究发现图 6.2 所对应的深度 SVM 网络整体上优于 SVM 网络(通过均方误差所衡量的损失函数大小),实际应用中,由于数据量级的限制,对于深度 SVM 网络增加层级、或进行数据扩张等策略与技巧,能否进一步提升网络的性能,需进一步研究。

为了使得由多个浅层网络堆栈形成的深度 SVM 网络奏效,通常激活(非线性)函数的选取为径向基函数,解决 SVM 两个缺点(一是模型的性能取决于先验选择的核函数;二是具有单层可调整的网络参数,其模型的表征能力有限)的深度 SVM 网络模型具有如下的优势:可有效地预防过拟合现象,可有效地根据支撑向量的个数自动确定模型的尺寸等。

备注:关于深度 SVM 的参考链接为 http://videolectures.net/roks2013_wiering_vector/.

6.2 深度 PCA 网络

一般认为,深度学习的核心在于自适应(层级)特征的提取,本节所描述的深度 PCA(主成分分析)网络,是指一种基于深度卷积神经网络架构的 PCA 参数初始化的无监督学习方式,其核心在于卷积流模块中卷积操作所使用的滤波器的学习。需要注意的是:该网络的模式仍为(无监督)预训练+(有监督)精调的半监督学习方式。本节的重点不在于对整个网络进行分析,仅对深度卷积神经网络中参数初始化阶段,滤波器的学习进行理解。下面从数据、滤波器学习等两个方面进行深度 PCA 网络的陈述。

1. 数据

$$\begin{cases} \{x^{(n)} \in \mathbb{R}^{32\times32@3}, \boldsymbol{y}^{(n)} \in \mathbb{R}^{10}\}_{n=1}^{N} \longrightarrow \text{Train Data} \\ \{x^{(t)} \in \mathbb{R}^{32\times32@3}\}_{t=1}^{T} \longrightarrow \text{Test Data} \end{cases} \tag{6.15}$$

数据集总的个数为 $N+T$。



2. 滤波器学习(参数初始化)

对于经典的深度卷积神经网络 LeNet 网络,见图 6.3,其中左侧实线框与右侧实线框为待学习的滤波器,其大小均为 5×5@64。

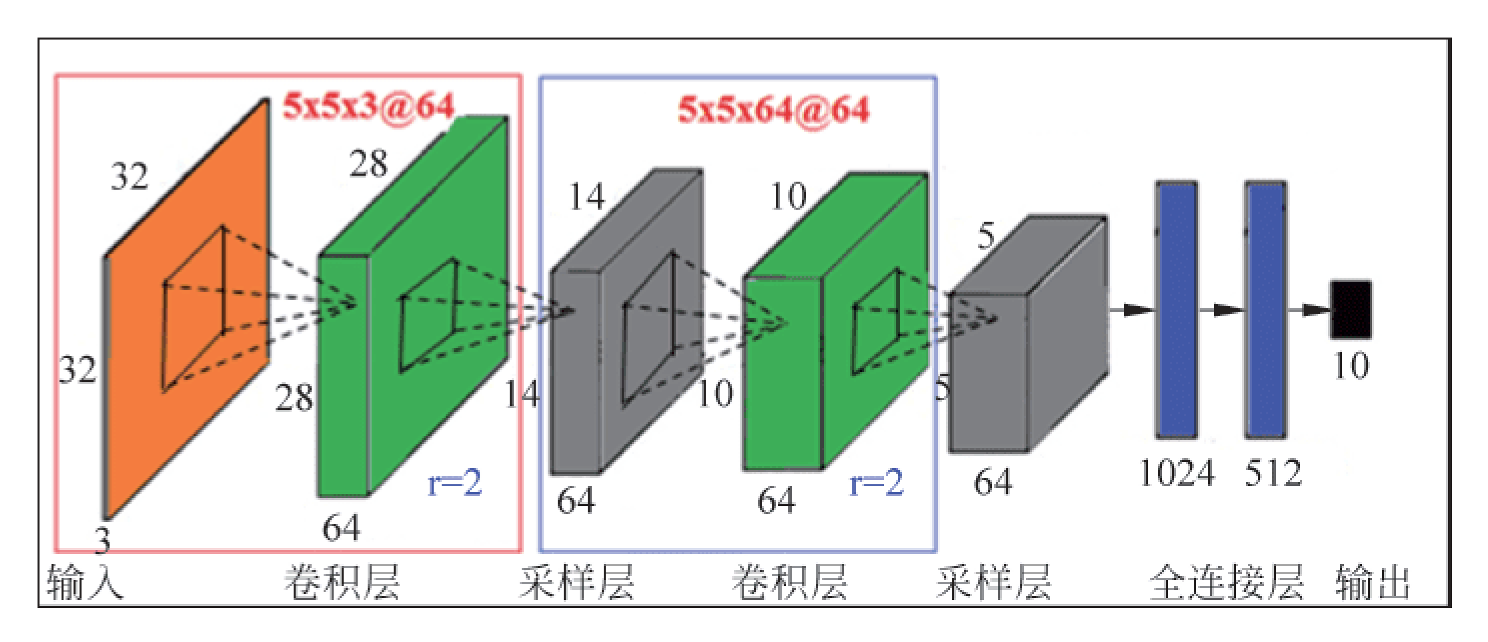


图 6.3 LeNet 网络结构

对于每一幅输入 x,由于滤波器的大小为 5×5×3,共计 64 个;所以,以窗口大小为 5×5×3 对输入进行滑块处理,得到 $(32-5+1)^2=784$ 个大小为 5×5×3 的块,将每一块先按行再按通道(三个通道)拉成一列,得到 75 维向量,最后对于输入 x,便得到相对应的一个矩阵 $\boldsymbol{M}(x)\in\mathbb{R}^{75\times784}$;由式(6.15)知道,测试数据集共有 T 幅,将这些所对应的矩阵堆栈起来,形成如下的矩阵:

$$\boldsymbol{M}=[\boldsymbol{M}(x^{(1)}),\cdots,\boldsymbol{M}(x^{(T)})]\in\mathbb{R}^{75\times(784\cdot T)} \tag{6.16}$$

然后,对矩阵 $\boldsymbol{M}$ 做完相应预处理后,进行奇异值分解,得到:

$$\boldsymbol{M}=\boldsymbol{U}\cdot\boldsymbol{\Sigma}\cdot\boldsymbol{V}^{\mathrm{T}} \tag{6.17}$$

根据奇异值 $\operatorname{diag}(\boldsymbol{\Sigma})$ 的大小次序,选择前 64 个特征值所对应的特征向量,并记为

$$\widetilde{\boldsymbol{W}}\in\mathbb{R}^{75\times64} \tag{6.18}$$

进一步,将其按通道再按行"逆向"得到滤波器组:

$$W^{(\text{layer1})}\in\mathbb{R}^{5\times5\times3@64} \tag{6.19}$$

即 64 个滤波器,每个滤波器的大小为 5×5×3,这个便作为输入层至第一隐层卷积操作中滤波器的参数初始化。

进一步利用池化后的第二个隐层作为输入 X,其大小为 14×14×64,注意此时输入的通道个数为 64,待学习的滤波器个数为 64,其大小为 5×5×64,每一幅输入对应的矩阵为:

$$\boldsymbol{M}(X)\in\mathbb{R}^{1600\times100} \tag{6.20}$$

同理,基于测试数据集,便可以得到如下的矩阵:

$$\boldsymbol{M}=[\boldsymbol{M}(X^{(1)}),\cdots,\boldsymbol{M}(X^{(T)})]\in\mathbb{R}^{1600\times(100\cdot T)} \tag{6.21}$$

同理,对该矩阵进行奇异值分解,得到其奇异值的前 64 个特征值所对应的特征向量,并按通道再按行"逆向"得到滤波器组:

$$W^{(\text{layer3})}\in\mathbb{R}^{5\times5\times64@64} \tag{6.22}$$

至此,便完成了基于深度卷积神经网络架构的 PCA 参数初始化的工作。后面的处理,



包括精调等，与深度卷积神经网络一致，这里不再赘述。

6.3　深度 ADMM 网络

众所周知，对于深度学习的研究，从模型设计方面，超参数(层数、隐层结点个数、激活函数等)的确定一直难以进行量化分析；从算法分析角度来看，模型的参数与数据量之间的关系，使得原本高度非线性的函数(即输入与输出之间的关系)易出现过拟合或欠拟合现象，进而导致模型学到的参数不收敛且出现梯度弥散的现象。什么是深度 ADMM 网络？如何解决超参数的设置问题以及算法的收敛特性？首先，ADMM 是一种非凸优化问题的求解方式，通过引入中间变量将正则项中参量从单目标函数中脱离处理，通过两个子问题(基于损失项的凸优化问题与基于正则项的软阈值求解)的交替求解来逼近原目标的解，与常见的 OMP 算法和 BP 算法不同。其次，为了结合深度学习与 ADMM 算法并实现各自优势互补，从而形成深度 ADMM 网络；该网络类似于深度反卷积神经网络，如何优化每一个模块的参数，即卷积稀疏编码下的合成滤波器与系数？使用的方法是 ADMM 算法。最后，从数据、模型、优化目标函数和求解四个方面对深度 ADMM 网络进行详述，先给出深度 ADMM 网络的架构，如图 6.4 所示。

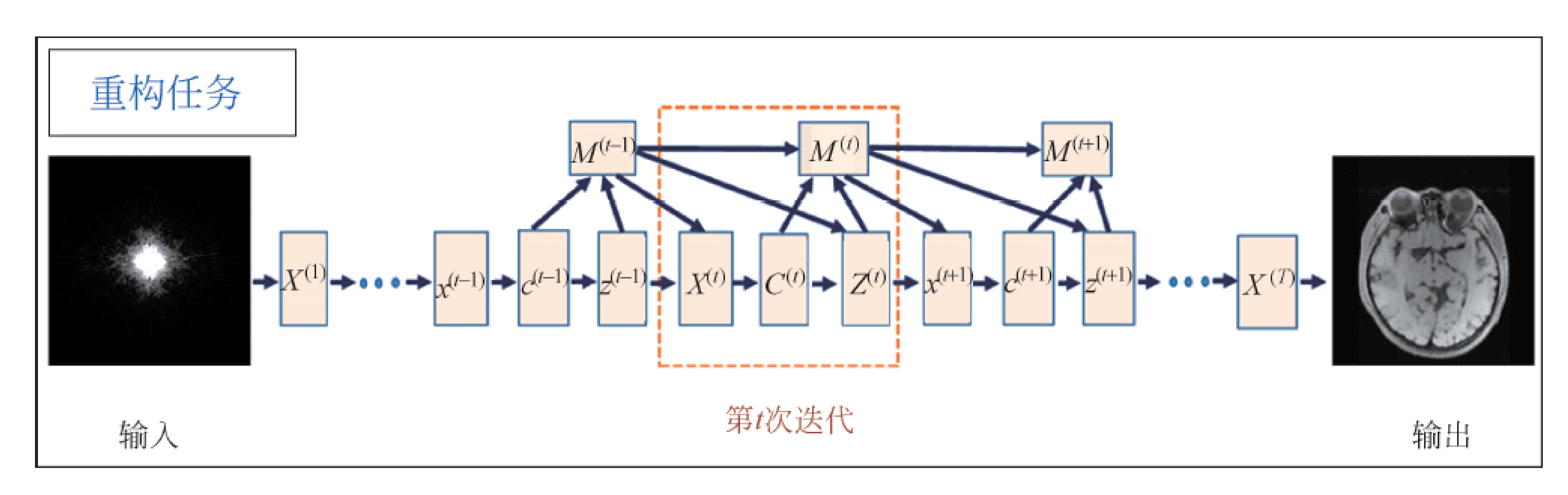


图 6.4　深度 ADMM 网络结构

1. 数据

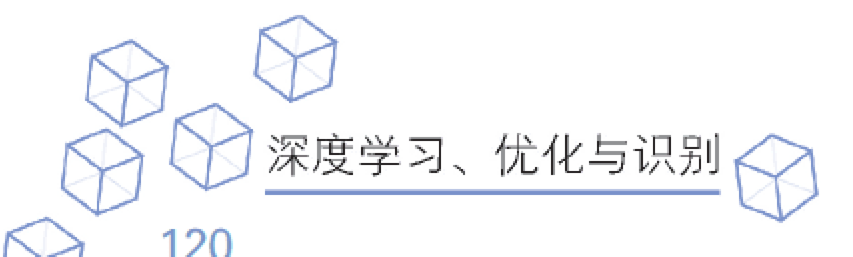
$$\{\boldsymbol{y}^{(n)} \in \mathbb{R}^m\}_{n=1}^N \tag{6.23}$$

注意，这里该网络用于重构任务，其中 $\boldsymbol{y}\in\mathbb{R}^m$ 是观测数据集，即对原始场景 $\boldsymbol{x}\in\mathbb{R}^s$ 通过观测矩阵$\boldsymbol{\Phi}\in\mathbb{R}^{m\times s}$获取，通常 $m<s$，重构任务的目的便是利用已知的观测数据集恢复或重构出原始场景。

备注：可以参考第 1 章稀疏表示相关知识点(压缩感知部分)。

2. 模型

$$\begin{cases}\boldsymbol{y}=\boldsymbol{\Phi}\cdot\boldsymbol{x}\\ \boldsymbol{\alpha}=\boldsymbol{D}\cdot\boldsymbol{x}\end{cases} \tag{6.24}$$



其中 $\boldsymbol{y}$ 的获取是通过观测矩阵$\boldsymbol{\Phi}$ 对场景的观测得到的，即数据的获取方式；另外为了导出原始场景中的某种先验，例如稀疏性等，即利用滤波器 $\boldsymbol{D}$ 得到原始场景的某种表示系数 α 来表征；需要注意的是$\boldsymbol{\Phi}\in\mathbb{R}^{u\times s}$中 $m<s$，而带来稀疏性的滤波器 $\boldsymbol{D}\in\mathbb{R}^{r\times s}$则有 $r>s$。通常，利用滤波器组来获取原始场景中的某种先验认知，例如$\{\boldsymbol{D}_l\in\mathbb{R}^{r\times s}\}_{l=1}^{L}$。

3. 优化目标函数

$$\begin{aligned}&\min_{\{x^{(n)}\},\boldsymbol{\Phi},\boldsymbol{D}} J(\{\boldsymbol{x}^{(n)}\}_{n=1}^{N},\boldsymbol{\Phi},\boldsymbol{D})\\&=\frac{1}{2\cdot N}\sum_{n=1}^{N}\left\{\|\boldsymbol{y}^{(n)}-\boldsymbol{\Phi}\cdot\boldsymbol{x}^{(n)}\|_2^2+\sum_{l=1}^{L}\lambda_l\cdot\rho(\boldsymbol{D}_l\cdot\boldsymbol{x}^{(n)})\right\}\end{aligned}\tag{6.25}$$

其中正则项中 $\lambda_l(l=1,2,\cdots,L)$为拉格朗日参数，$D_l(l=1,2,\cdots,L)$为 L 个滤波器，$\rho(\cdot)$为某种正则函数，例如关于稀疏性的 $L_p(0\leqslant p\leqslant 1)$范数等；损失项中$\boldsymbol{\Phi}$ 为观测矩阵。

假设观测矩阵和滤波器组都已给定，不再需要训练学习获取；所以对于观测数据 $\boldsymbol{y}$，其目标函数可写为：

$$\min_{x}\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}\cdot\boldsymbol{x}\|_2^2+\sum_{l=1}^{L}\lambda_l\cdot\rho(\boldsymbol{D}_l\cdot\boldsymbol{x})\tag{6.26}$$

4. 求解

这里重点求解的是优化目标(6.26)，利用 ADMM 算法，通过引入中间变量 $\boldsymbol{z}_l=\boldsymbol{D}_l\cdot\boldsymbol{x}$，故目标函数可以进一步写为：

$$\begin{aligned}&\min_{x}\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}\cdot\boldsymbol{x}\|_2^2+\sum_{l=1}^{L}\lambda_l\cdot\rho(\boldsymbol{z}_l)\\&\text{s. t. }\boldsymbol{z}_l=\boldsymbol{D}_l\cdot\boldsymbol{x},\quad l=1,2,\cdots,L\end{aligned}\tag{6.27}$$

转化为无约束目标函数：

$$\min_{\boldsymbol{x},\{\boldsymbol{z}_l,\boldsymbol{\beta}_l\}}\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}\cdot\boldsymbol{x}\|_2^2+\sum_{l=1}^{L}\lambda_l\cdot\rho(\boldsymbol{z}_l)-\sum_{l=1}^{L}\langle\boldsymbol{\beta}_l,\boldsymbol{z}_l-\boldsymbol{D}_l\cdot\boldsymbol{x}\rangle+\sum_{l=1}^{L}\frac{\gamma_l}{2}\|\boldsymbol{z}_l-\boldsymbol{D}_l\cdot\boldsymbol{x}\|_2^2\tag{6.28}$$

迭代优化求解公式为：

$$\begin{cases}\boldsymbol{x}^{(t+1)}=\arg\min_{\boldsymbol{x}}\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{\Phi}\cdot\boldsymbol{x}\|_2^2-\sum_{l=1}^{L}\langle\boldsymbol{\beta}_l^{(t)},\boldsymbol{z}_l^{(t)}-\boldsymbol{D}_l\cdot\boldsymbol{x}\rangle+\sum_{l=1}^{L}\frac{\gamma_l}{2}\|\boldsymbol{z}_l^{(t)}-\boldsymbol{D}_l\cdot\boldsymbol{x}\|_2^2\\\boldsymbol{z}^{(t+1)}=\arg\min_{\boldsymbol{z}}\sum_{l=1}^{L}\lambda_l\cdot\rho(\boldsymbol{z}_l)-\sum_{l=1}^{L}\langle\boldsymbol{\beta}_l^{(t)},\boldsymbol{z}_l-\boldsymbol{D}_l\cdot\boldsymbol{x}^{(t+1)}\rangle+\sum_{l=1}^{L}\frac{\gamma_l}{2}\|\boldsymbol{z}_l-\boldsymbol{D}_l\cdot\boldsymbol{x}^{(t+1)}\|_2^2\\\boldsymbol{\beta}^{(t+1)}=\arg\min_{\boldsymbol{\beta}}\sum_{l=1}^{L}\langle\boldsymbol{\beta}_l,\boldsymbol{z}_l^{(t+1)}-\boldsymbol{D}_l\cdot\boldsymbol{x}^{(t+1)}\rangle\end{cases}\tag{6.29}$$

注意 t 为第 t 次迭代，另外 $\boldsymbol{z}^{(t)}$ 和$\boldsymbol{\beta}^{(t)}$ 简记为：

$$z^{(t)} \triangleq \{z_l^{(t)}\}_{l=1}^{L}, \boldsymbol{\beta}^{(t)} \triangleq \{\boldsymbol{\beta}_l^{(t)}\}_{l=1}^{L} \tag{6.30}$$

进一步，关于式(6.29)有：

$$\begin{cases} \boldsymbol{x}^{(t)} = \left[\boldsymbol{\Phi}^{\mathrm{T}} \cdot \boldsymbol{\Phi} - \sum_{l=1}^{L} \gamma_l \cdot (\boldsymbol{D}_l)^{\mathrm{T}} \boldsymbol{D}_l\right]^{-1} \left[\boldsymbol{\Phi}^{\mathrm{T}} \cdot \boldsymbol{y} - \sum_{l=1}^{L} (\boldsymbol{D}_l)^{\mathrm{T}} \cdot (\boldsymbol{\beta}_l^{(t-1)} + \gamma_l \cdot \boldsymbol{z}_l^{(t-1)})\right] \\ \boldsymbol{z}^{(t)} : \boldsymbol{z}_l^{(t)} = S\left(\boldsymbol{D}_l \cdot \boldsymbol{x}^{(t)} + \boldsymbol{\beta}_l^{(t-1)}; \frac{\lambda_l}{\gamma_l}\right), l = 1, 2, \cdots, L \\ \boldsymbol{\beta}^{(t)} : \boldsymbol{\beta}_l^{(t)} = \boldsymbol{\beta}_l^{(t)} + \eta \cdot (\boldsymbol{z}_l^{(t)} - \boldsymbol{D}_l \cdot \boldsymbol{x}^{(t)}) \end{cases} \tag{6.31}$$

其中 η 为学习速率，这样便可以交替更新直至收敛，求出原始场景 $\boldsymbol{x}$，即

$$\lim_{t \to \infty} \boldsymbol{x}^{(t)} = \boldsymbol{x} \tag{6.32}$$

接下来，为了与深度卷积神经网络建立相应的操作分析关系，类比卷积、非线性、池化等操作，基于式(6.31)分析 ADMM 策略含义如下：首先，卷积层为线性处理后得到若干特征图，记为模块 $X^{(t)}$，即 $\boldsymbol{x}^{(t)}$ 的更新，本质上为线性操作。其次非线性处理(激活函数)，记为模块 $C^{(t)}$，其中收缩函数 $S(\cdot)$ 为非线性函数，即模块 $X^{(t)}$ 到 $Z^{(t)}$ 的非线性映射。最后参数更新层，即模块 $M^{(t)}$ 包括 $\boldsymbol{\beta}^{(t)}$，学习速率 η 等。另外需要注意的是网络结构中的“深度”是指更新的迭代次数，即见图 6.3。

6.4　深度极限学习机

6.4.1　极限学习机

极限学习机也称超限学习机，它是一种新型的快速学习算法，对于单隐层神经网络，它可以随机初始化输入层与隐层之间的权值连接矩阵和偏置，并通过学习或训练得到隐层到输出层之间的权值矩阵。下面给出其网络结构，如图 6.5 所示。

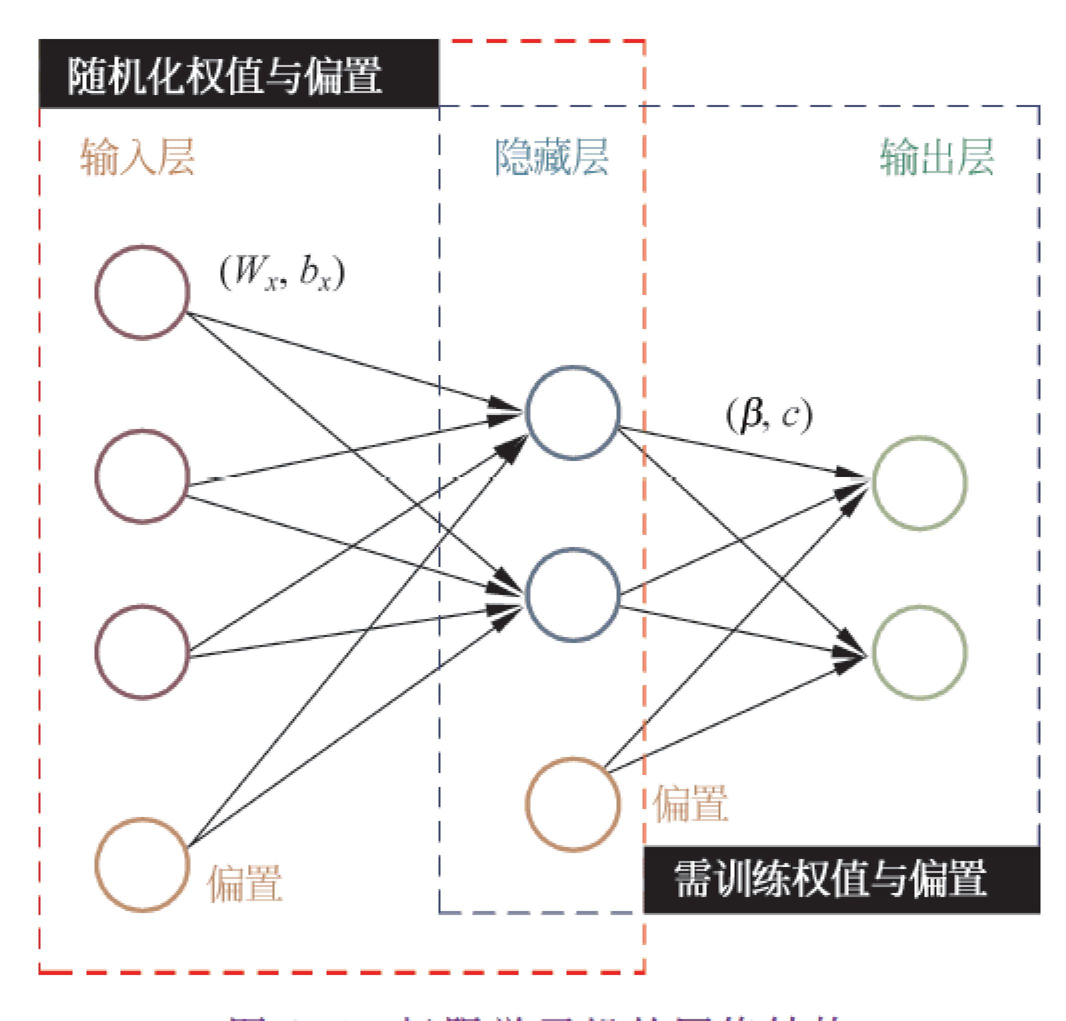


图 6.5　极限学习机的网络结构

另外，极限学习机的优势在于：只需要设置网络的隐层节点个数，在算法执行过程中不需要调整网络的输入层到隐层的权值及偏置，并产生唯一的最优解，具有学习速度快且泛化性能好的优点。

1. 数据

$$\{\boldsymbol{x}^{(n)} \in \mathbb{R}^m, \boldsymbol{y}^{(n)} \in \mathbb{R}^s\}_{n=1}^N \tag{6.33}$$

2. 模型

$$\begin{cases} \boldsymbol{h}_x = \sigma(\boldsymbol{W}_x \cdot \boldsymbol{x} + \boldsymbol{b}_x) \in \mathbb{R}^r \\ \boldsymbol{y} = \boldsymbol{\beta} \cdot \boldsymbol{h}_x + \boldsymbol{c} \in \mathbb{R}^s \end{cases} \tag{6.34}$$

注意随着输入的不同，随机化权值与偏置也发生变化，即对于输入 $\boldsymbol{x}^{(1)} \neq \boldsymbol{x}^{(2)}$，则有：

$$\begin{cases} \boldsymbol{W}_{x^{(1)}} \neq \boldsymbol{W}_{x^{(2)}} \\ \boldsymbol{b}_{x^{(1)}} \neq \boldsymbol{b}_{x^{(2)}} \end{cases} \tag{6.35}$$

同时，权值与偏置$(\boldsymbol{W}_x, \boldsymbol{b}_x)$不需要学习，服从某种分布下的随机化获取，待学习的参数为$(\boldsymbol{\beta}, \boldsymbol{c})$。其中隐层节点的个数为 r，所以$(\boldsymbol{W}_x \in \mathbb{R}^{r\times u}, \boldsymbol{b}_x \in \mathbb{R}^r)$，$\boldsymbol{\beta} \in \mathbb{R}^{s\times r}$ 和 $\boldsymbol{c} \in \mathbb{R}^s$；$\sigma(\cdot)$为激活函数。

3. 优化目标函数

$$\min_{(\boldsymbol{\beta},\boldsymbol{c})} \frac{1}{2} \sum_{n=1}^{N} \| \boldsymbol{y}^{(n)} - \hat{\boldsymbol{y}}^{(n)} \|_2^2 \tag{6.36}$$

将其展开，并令 $\boldsymbol{y}^{(n)} = \hat{\boldsymbol{y}}^{(n)} (n=1,2,\cdots,N)$，得到：

$$\boldsymbol{\beta} \cdot \boldsymbol{H} + \boldsymbol{C} = \boldsymbol{Y} \tag{6.37}$$

其中隐层特征为：

$$\boldsymbol{H} = [\boldsymbol{h}_{x^{(1)}}, \boldsymbol{h}_{x^{(2)}}, \cdots, \boldsymbol{h}_{x^{(N)}}] \in \mathbb{R}^{r\times N} \tag{6.38}$$

其中 $r<N$；另外，符号 $\boldsymbol{C}$ 和期望输出 $\boldsymbol{Y}$ 为：

$$\begin{cases} \boldsymbol{C} = [\boldsymbol{c}, \boldsymbol{c}, \cdots, \boldsymbol{c}] \in \mathbb{R}^{s\times N} \\ \boldsymbol{Y} = [\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \cdots, \boldsymbol{y}^{(N)}] \in \mathbb{R}^{s\times N} \end{cases} \tag{6.39}$$

4. 求解

接下来，对式(6.37)利用 Moore-Penrose 广义逆，通常为了计算方便，将隐层到输出层之间的偏置设置为 $\boldsymbol{C}=\boldsymbol{0} \in \mathbb{R}^{s\times N}$，故得到：

$$\boldsymbol{\beta} = \boldsymbol{Y} \cdot \boldsymbol{H}^{\dagger} \tag{6.40}$$

这里 $\boldsymbol{H}^{\dagger} = \boldsymbol{H}^{\mathrm{T}}(\boldsymbol{H} \cdot \boldsymbol{H}^{\mathrm{T}})^{-1}$；已经证明，一旦输入到隐层之间的权值矩阵和隐层偏置被随机确定，则隐层的输出矩阵 $\boldsymbol{H}$ 就被唯一确定，求得的$\boldsymbol{\beta}$是满足优化目标函数(6.36)最小的、且唯一的解。

注意一：若 $\boldsymbol{H} \cdot \boldsymbol{H}^{\mathrm{T}} \in \mathbb{R}^{r\times r}$不可逆，则需要加正则项，得到：



$$\boldsymbol{H}^{\dagger}=\boldsymbol{H}^{\mathrm{T}}\left(\boldsymbol{H}\cdot\boldsymbol{H}^{\mathrm{T}}+\frac{1}{\lambda}\cdot\boldsymbol{I}_r\right)^{-1} \tag{6.41}$$

其中 λ 为拉格朗日因子，即给优化目标函数(6.36)加入如下的正则项，得到：

$$\min_{(\boldsymbol{\beta},c)}\frac{\lambda}{2}\sum_{n=1}^{N}\|\boldsymbol{y}^{(n)}-\hat{\boldsymbol{y}}^{(n)}\|_2^2+\frac{1}{2}\|\boldsymbol{\beta}\|_F^2 \tag{6.42}$$

注意二：核极限学习机，是指跳过随机化权值矩阵与偏置根据输入得到隐层输出的过程，即式(6.34)中的：

$$\boldsymbol{h}_x=\sigma(\boldsymbol{W}_x\cdot\boldsymbol{x}+\boldsymbol{b}_x) \tag{6.43}$$

依据输入与输出之间的关系，且 $r>N$，即隐层节点个数大于样本的个数：

$$\boldsymbol{y}=\boldsymbol{\beta}\cdot\boldsymbol{h}_x=\boldsymbol{Y}\cdot\boldsymbol{H}^{\dagger}\cdot\boldsymbol{h}_x=\boldsymbol{Y}\cdot(\boldsymbol{H}^{\mathrm{T}}\cdot\boldsymbol{H})^{-1}\cdot\boldsymbol{H}^{\mathrm{T}}\cdot\boldsymbol{h}_x \tag{6.44}$$

注意这里的伪逆，由于 $r>N$，变为：

$$\boldsymbol{H}^{\dagger}=(\boldsymbol{H}^{\mathrm{T}}\cdot\boldsymbol{H})^{-1}\cdot\boldsymbol{H}^{\mathrm{T}} \tag{6.45}$$

引入核函数，使得：

$$\boldsymbol{H}^{\mathrm{T}}\cdot\boldsymbol{H}=\begin{bmatrix}\kappa(\boldsymbol{x}_1,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_1,\boldsymbol{x}_N)\\ \vdots & \ddots & \vdots\\ \kappa(\boldsymbol{x}_N,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_N,\boldsymbol{x}_N)\end{bmatrix}\in\mathbb{R}^{N\times N} \tag{6.46}$$

其中 $\kappa(\cdot,\cdot)$ 为核函数，进而有：

$$\boldsymbol{H}^{\mathrm{T}}\cdot\boldsymbol{h}_x=\begin{bmatrix}\kappa(\boldsymbol{x}_1,\boldsymbol{x})\\ \vdots\\ \kappa(\boldsymbol{x}_N,\boldsymbol{x})\end{bmatrix}\in\mathbb{R}^{N\times 1} \tag{6.47}$$

所以，根据式(6.44)输入输出之间的公式可写为：

$$\boldsymbol{y}=\boldsymbol{Y}\cdot\left(\begin{bmatrix}\kappa(\boldsymbol{x}_1,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_1,\boldsymbol{x}_N)\\ \vdots & \ddots & \vdots\\ \kappa(\boldsymbol{x}_N,\boldsymbol{x}_1) & \cdots & \kappa(\boldsymbol{x}_N,\boldsymbol{x}_N)\end{bmatrix}\right)^{-1}\cdot\begin{bmatrix}\kappa(\boldsymbol{x}_1,\boldsymbol{x})\\ \vdots\\ \kappa(\boldsymbol{x}_N,\boldsymbol{x})\end{bmatrix} \tag{6.48}$$

6.4.2　深度极限学习机

深度极限学习机的基本模块是基于极限学习机的自编码网络，所以下面来详述深度极限学习机。

1. 基于极限学习机的自编码网络

该网络结构(图 6.6)的核心是学习参数 β，输入与期望输出是一致的，所以是一种无监督的学习方式，其过程类似于 6.4.1 节，不过将输入替代为输出，即式(6.33)中数据变为：

$$\{\boldsymbol{x}^{(n)}\in\mathbb{R}^m\}_{n=1}^N \tag{6.49}$$

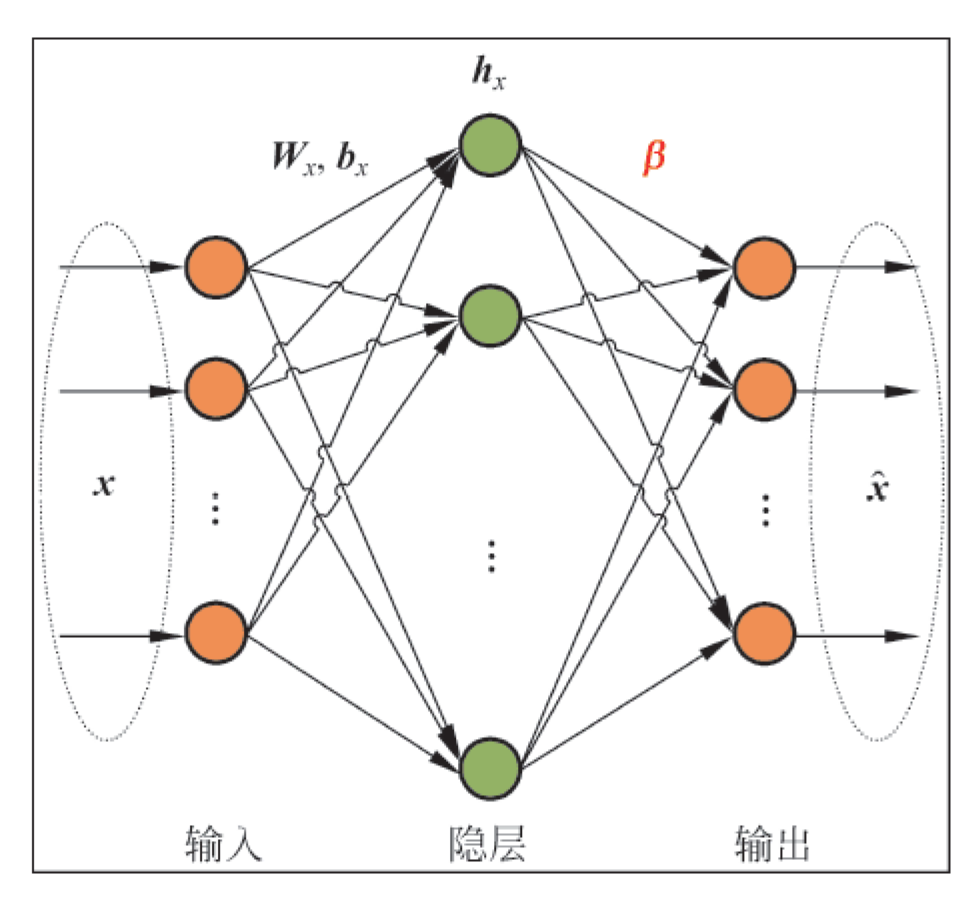


图 6.6　基于极限学习机的自编码网络结构

其后关于参数 β 的求解过程，与之前一致，这里不再赘述。

备注：关于自编码网络的详细描述参考第 4 章。

2. 深度极限学习机

下面基于极限学习机的自编码网络，通过逐层堆栈的方式形成深度极限学习机，其网络结构的模型如图 6.7 所示。

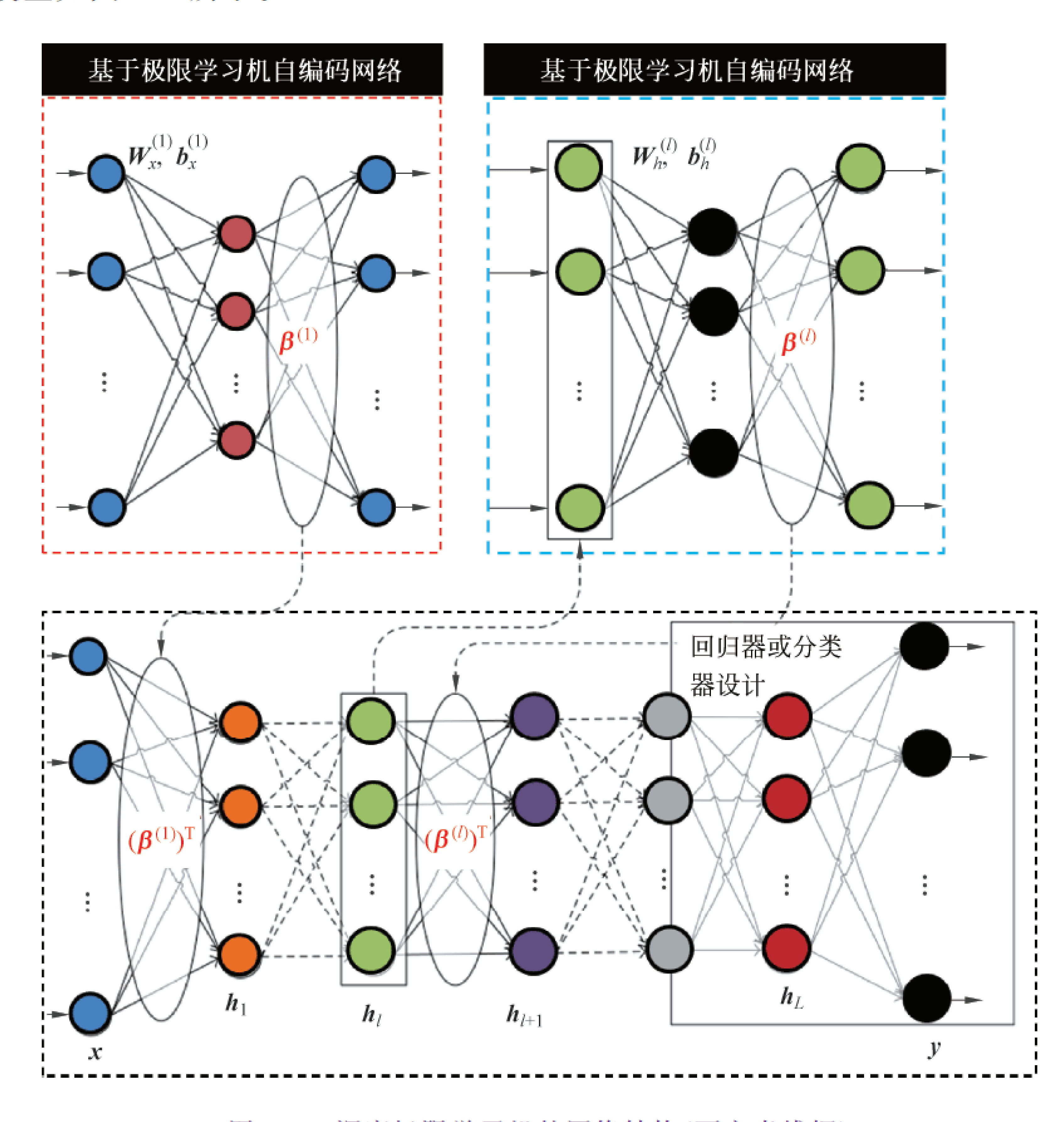


图 6.7　深度极限学习机的网络结构(下方虚线框)

下面从数据、模型、优化目标函数和求解四个方面来详述和理解该网络的结构特性。

1）数据

$$\begin{cases} \{\boldsymbol{x}^{(n)} \in \mathbb{R}^m, \boldsymbol{y}^{(n)} \in \mathbb{R}^s\}_{n=1}^{N} \longrightarrow \text{Train Data} \\ \{\boldsymbol{x}^{(t)} \in \mathbb{R}^m\}_{t=1}^{T} \longrightarrow \text{Test Data} \end{cases} \tag{6.50}$$

其中测试数据集用于学习层级间的参数(类似逐层学习参数初始化)，训练数据集用于精调整个网络，特别是深度极限学习机中的回归器或分类器的设计。

2）模型

$$\begin{cases} \boldsymbol{h}_l = \sigma(\boldsymbol{\beta}^{(l)} \cdot \boldsymbol{h}_{l-1}) \\ \boldsymbol{y} = \boldsymbol{\beta}_C \cdot \boldsymbol{h}_L \end{cases} \tag{6.51}$$

其中 $l=1,2,\cdots,L$，且 $\boldsymbol{h}_0=\boldsymbol{x}$，参数 $\{\boldsymbol{\beta}^{(l)}\}_{l=1}^{L}$ 可以通过无监督学习方式下的极限学习机自编码网络学习得到，另外 $\sigma(\cdot)$ 为激活函数，$\boldsymbol{\beta}_C$ 为最后回归器或分类器设计阶段的参数，即若已知第 L 个隐层的特征 $\boldsymbol{h}_L$，将其视为极限学习机的输入，与输出 $\boldsymbol{y}$ 建立关系，待学习的参数便为 $\boldsymbol{\beta}_C$。

3）优化目标函数

$$\min_{\{\boldsymbol{\beta}^{(l)}\},\boldsymbol{\beta}_C} \frac{1}{N}\sum_{n=1}^{N} \| \hat{\boldsymbol{y}}^{(n)} - \boldsymbol{y}^{(n)} \|_2^2 + \lambda \sum_{l=1}^{L} \| \boldsymbol{\beta}^{(l)} \|_F^2 + \gamma \| \boldsymbol{\beta}_C \|_F^2 \tag{6.52}$$

其中参数初始化的过程（测试数据集上进行无监督学习）采用如下的目标函数：

$$\min_{\boldsymbol{\beta}^{(l)}} \frac{1}{T}\sum_{t=1}^{T} \| \boldsymbol{h}_t^{(l)} - \hat{\boldsymbol{h}}_t^{(l)} \|_2^2 + \lambda \cdot \| \boldsymbol{\beta}^{(l)} \|_F^2 \tag{6.53}$$

其中：

$$\boldsymbol{h}_t^{(l)} = \boldsymbol{\beta}^{(l)} \cdot \sigma(\boldsymbol{W}^{(l)} \cdot \boldsymbol{h}_t^{(l-1)} + \boldsymbol{b}^{(l)}) \tag{6.54}$$

公式中的 $\sigma(\cdot)$ 为激活函数，参数 $(\boldsymbol{W}^{(l)}, b^{(l)})$ 为第 $l-1$ 个隐层到第 l 个隐层之间的随机化初始权值矩阵与偏置（服从某种概率分布下的采样），注意 $\boldsymbol{h}_t^{(0)}=\boldsymbol{x}^{(t)}$。

4）求解

关于式(6.53)的求解参考极限学习机的求解部分，这里不再赘述；另外关于整个深度极限学习机的网络精调，利用随机梯度下降的方式进行。

综上所述，深度极限学习机可以提取出数据中高层次的抽象信息；极限学习机的理论避免了网络权值的反复迭代调整，提高了计算效率；另外，半监督的逐层训练机制解决了很多实际问题中训练标签难以获取的问题。因此，深度极限学习机能同时有效地解决大数据时代高维度，异构数据，获取标记样本难、构造特征难、训练难等问题。

备注：更多关于深度或多层极限学习机的理论、代码与应用可以参考黄广斌的个人网页：http://www.ntu.edu.sg/home/egbhuang/elm_codes.html.

6.5　深度多尺度几何网络

深度多尺度几何网络是指基于三代小波（例如脊波、曲波、轮廓波、楔形波等）所构建的深度神经网络，与传统的基于一代小波（例如 Meyer 小波、Morlet 小波、Haar 小波、Gaussian 小波、Doubechies 小波系列等）所构建的深度神经网络主要不同之处在于：一是几乎不再使用（三代）小波作为激活函数，来调整隐层线性输出后的扭曲程度或响应特性，更多地使用在权值矩阵初始化、多尺度或多通路网络的设计中，以避免过早地陷入局部最优，以



及获取同一层级不同分辨率下(或角度下)的特征来提升网络表征能力。二是三代小波相比一代小波更易获取输入场景(图像)多频带、多角度下的拓扑(几何结构)特性描述,类比于生物神经系统,腹侧视觉通路不仅能够回答看到的事物是什么这个生物学功能,而且这种生物学功能中还蕴含着不依赖事物的远近(数学上描述为分辨率)、角度等,深度神经网络恰好能够模拟这个功能,而它与三代小波(多尺度几何分析)的结合更易体现蕴含着的特性。这里,我们仅对深度脊波网络、深度轮廓波网络作以详细的描述。

6.5.1 深度脊波网络

为了方便下面的论述,首先,我们给出脊波的定义:若 $\psi(x)\in L^2(\mathbb{R})$ 满足容许性条件,即

$$C_\psi=\int_{\mathbb{R}}\left(\frac{|\hat{\psi}(\omega)|^2}{|\omega|^d}\right)\mathrm{d}\omega<+\infty \tag{6.55}$$

其中:

$$\hat{\psi}(\omega)=\int_{\mathbb{R}}\psi(x)\mathrm{e}^{-\mathrm{j}\omega x}\mathrm{d}(x) \tag{6.56}$$

称 $\psi(x)$ 为容许性神经激活函数;进一步,由该函数产生的脊函数:

$$\begin{cases}\psi_{(a,u,b)}(x)=\dfrac{1}{\sqrt{a}}\psi\left(\dfrac{u\cdot x-b}{a}\right)\\ a\in\mathbb{R}^+,b\in\mathbb{R},u\in S^{d-1},\|u\|=1\end{cases} \tag{6.57}$$

为脊波,其中 S^{d-1} 为单位球,符号 (a,b,u) 分别表示尺度因子、平移因子和方向因子。

通常将神经网络中的激活函数(例如 Sigmoid 函数等)用脊波来替代形成脊波神经网络,其结构如图 6.8 所示。

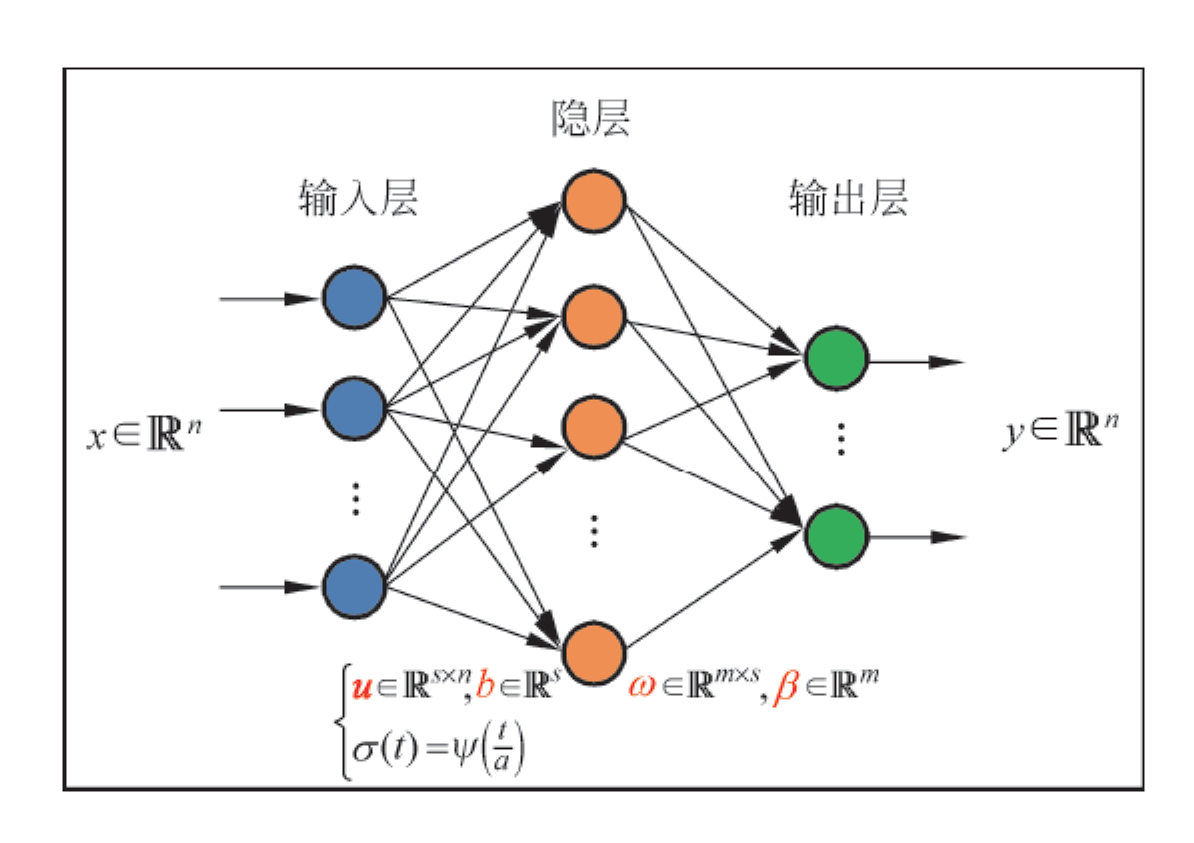


图 6.8 脊波神经网络结构

1. 数据

$$\{\boldsymbol{x}^{(t)} \in \mathbb{R}^{n}, \boldsymbol{y}^{(t)} \in \mathbb{R}^{m}\}_{t=1}^{T} \tag{6.58}$$

2. 模型

$$\begin{cases} \psi\left(\dfrac{\boldsymbol{u} \cdot \boldsymbol{x} - \boldsymbol{b}}{\boldsymbol{a}}\right) = \left(\psi\left(\dfrac{\boldsymbol{u}(1,:) \cdot \boldsymbol{x} - b(1)}{a(1)}\right), \cdots, \psi\left(\dfrac{\boldsymbol{u}(s,:) \cdot \boldsymbol{x} - b(s)}{a(s)}\right)\right)^{T} \in \mathbb{R}^{s} \\ \boldsymbol{h} = \sigma(\boldsymbol{u} \cdot x - b) = \psi\left(\dfrac{\boldsymbol{u} \cdot \boldsymbol{x} - \boldsymbol{b}}{\boldsymbol{a}}\right) \\ \boldsymbol{y} = \boldsymbol{\omega} \cdot \boldsymbol{h} + \boldsymbol{\beta} \in \mathbb{R}^{m} \end{cases} \tag{6.59}$$

模型中待优化的参数为：

$$\theta = [\boldsymbol{u}, \boldsymbol{b}, \boldsymbol{a}; \boldsymbol{\omega}, \boldsymbol{\beta}] \tag{6.60}$$

3. 优化目标函数

$$\min_{\theta} J(\theta) = \frac{1}{2T}\sum_{t=1}^{T} \| \hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)} \|_{2}^{2} + \frac{\lambda}{2} \cdot \| \boldsymbol{u} \|_{F}^{2} + \frac{\gamma}{2} \| \boldsymbol{\omega} \|_{F}^{2} \tag{6.61}$$

4. 求解

按照梯度下降法可以求出相应的参数更新公式，即

$$\begin{cases} \boldsymbol{\omega}^{(k+1)} = \boldsymbol{\omega}^{(k)} - \eta \cdot ?\boldsymbol{\omega} \mid_{\boldsymbol{\omega}=\boldsymbol{\omega}^{(k)}} \\ \boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} - \eta \cdot ?\boldsymbol{\beta} \mid_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(k)}} \\ \boldsymbol{a}^{(k+1)} = \boldsymbol{a}^{(k)} - \eta \cdot ?\boldsymbol{a} \mid_{\boldsymbol{a}=\boldsymbol{a}^{(k)}} \\ \boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} - \eta \cdot ?\boldsymbol{u} \mid_{\boldsymbol{u}=\boldsymbol{u}^{(k)}} \\ \boldsymbol{b}^{(k+1)} = \boldsymbol{b}^{(k)} - \eta \cdot ?\boldsymbol{b} \mid_{\boldsymbol{b}=\boldsymbol{b}^{(k)}} \end{cases} \tag{6.62}$$

其中 η 为学习速率，梯度下降量分别为：

$$\begin{cases} ?\boldsymbol{\omega} \mid_{\boldsymbol{\omega}=\boldsymbol{\omega}^{(k)}} = \displaystyle\sum_{t=1}^{T} (\hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)}) \cdot (\boldsymbol{h}^{(t)})^{T} + \gamma \cdot \boldsymbol{\omega} \Big|_{\boldsymbol{\omega}=\boldsymbol{\omega}^{(k)}} \in \mathbb{R}^{m\times s} \\ ?\boldsymbol{\beta} \mid_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(k)}} = \displaystyle\sum_{t=1}^{T} (\hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)}) \Big|_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(k)}} \in \mathbb{R}^{m} \end{cases} \tag{6.63}$$

进一步，通过引入隐层的误差传播项，利用链式法则便可以求解参数($\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{u}$)的梯度下降量。这里不再赘述。

综上所述，根据三层的脊波神经网络可以利用自编码的方式通过逐层堆栈的方式形成深度脊波神经网络。该模型的优点：一是半监督的学习方式(无监督逐层参数初始化加有监督整体网络精调)；二是融入脊波的特性，即灵活的结构和快速并行的处理速度，以及较强的容错性和鲁棒性等。

6.5.2　深度轮廓波网络

自从轮廓波被提出以后，探索轮廓波变换后的数据特征，已发现该变换能够带来稀疏

性、多尺度特性、多方向性、局部化、低冗余度、平移不变性、容易实现和计算高效等众多的性质，能否将其与深度神经网络进行结合，充分发挥二者各自的优势？为了探索研究该问题，首先，轮廓波变换的核心思想是在多尺度的基础上实现方向信息的获取，其变换所对应的滤波器分为拉普拉斯塔式分解和方向滤波器组两部分，其中拉普拉斯金字塔滤波器主要完成奇异点的分离任务；方向滤波器主要完成奇异点的收集工作，即利用方向基本相同的准则，将奇异点收集到一个基函数上集中描述，常用轮廓波的变换有非下采样轮廓波变换、全相位轮廓波变换、基于小波的轮廓波变换、抗混叠轮廓波变换以及复轮廓波变换等。其次，本节主要描述非下采样轮廓波变换与深度神经网络的结合，给出深度轮廓波网络的整体结构。

1. 非下采样轮廓波变换

如图 6.9 所示，对于输入图像，利用拉普拉斯金字塔分解(分解阶段包含两个滤波器，一个是低通滤波器(主要获取输入中的低频成分)；另一个是高通滤波器(主要获取高频成分))；进一步利用方向滤波器组对高频成分进行滤波，其中方向为人工设置(如图中设置为 8 个方向)；那么经过一级分解过后的变换系数为 8 个方向下的高频成分和一个低频成分；如果进行两级分解，只需对低频成分如是操作，即将低频成分视为第二级分解的输入，注意由于是非下采样，所以得到的所有变换系数的尺寸与输入一致；类似于小波分解，只对低频成分进行拉普拉斯金字塔分解，对高频成分进行不同方向下的滤波处理。值得指出的是，非下采样的轮廓波变换与轮廓波变换的区别主要在于是否进行下采样的操作，若进行下采样操作，则随着层级的分解在低频成分上进行操作。另外，不论是拉普拉斯金字塔分解还是方向滤波器，其滤波器的构造独立于输入信号，即所有的滤波器事先确定。

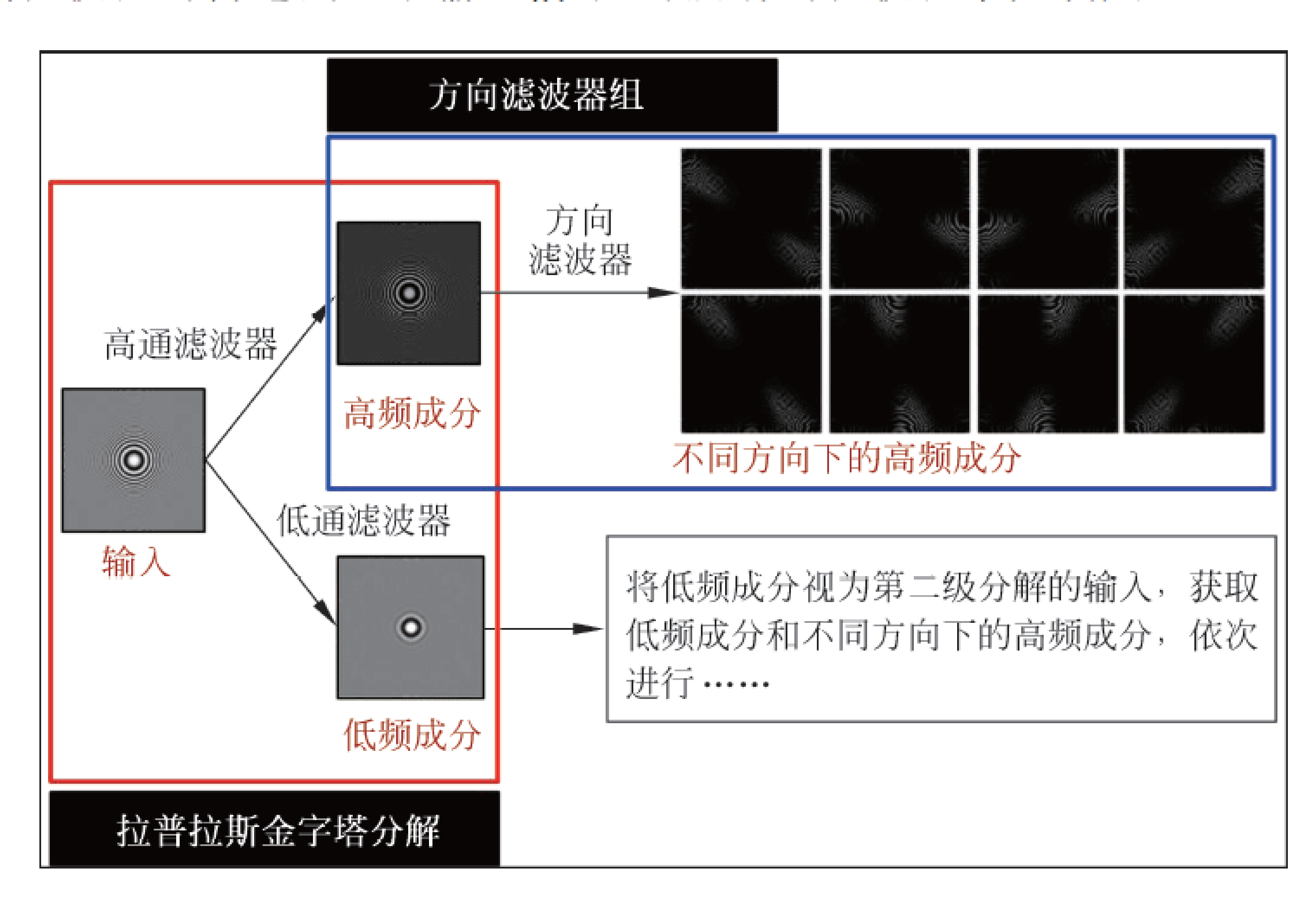


图 6.9　非下采样轮廓波变换(一级分解)

非下采样轮廓波变换所对应的数学公式为：一级分解，分为两个阶段，一是拉普拉斯塔式分解：

$$\begin{cases} \boldsymbol{x}_H^{(1)} = \boldsymbol{x} * PF_H^{(D)} \in \mathbb{R}^{n\times m} \\ \boldsymbol{x}_L^{(1)} = \boldsymbol{x} * PF_H^{(D)} \in \mathbb{R}^{n\times m} \end{cases} \tag{6.64}$$

其中 $\boldsymbol{x}\in\mathbb{R}^{n\times m}$ 为输入信号，$PF_H^{(D)}$ 为分解(Decomposition)阶段的高通滤波器，$\boldsymbol{x}_H^{(1)}$ 为一级分解后的高频成分；$PF_L^{(D)}$ 为低通滤波器，$\boldsymbol{x}_L^{(1)}$ 为一级分解后的低频成分，通常也记输入 $\boldsymbol{x}_L^{(0)}=\boldsymbol{x}$；二是方向滤波器组：

$$\begin{cases} \boldsymbol{x}_{H,1}^{(1)} = \boldsymbol{x}_H^{(1)} * DF_1 \in \mathbb{R}^{n\times m} \\ \boldsymbol{x}_{H,2}^{(1)} = \boldsymbol{x}_H^{(1)} * DF_2 \in \mathbb{R}^{n\times m} \\ \vdots \\ \boldsymbol{x}_{H,K}^{(1)} = \boldsymbol{x}_H^{(1)} * DF_K \in \mathbb{R}^{n\times m} \end{cases} \tag{6.65}$$

其中 $DF_k(k=1,2,\cdots,K)$ 为方向滤波器组，通常 K 为取 2 的指数级，那么对输入 $\boldsymbol{x}_L^{(0)}$ 进行一级非下采样轮廓波分解后的变换系数为：

$$X^{(1)} = [\{\boldsymbol{x}_{H,k}^{(1)}\}_{k=1}^{K}, \boldsymbol{x}_L^{(1)}] \tag{6.66}$$

若对输入 $x_L^{(0)}$ 进行两级分解，则只需继续对 $x_L^{(1)}$ 再进行一次一级分解即可，需要注意的是：这两级中，每一级上对高频成分进行方向滤波的个数可能是不同的。

2. 深度轮廓波网络

若利用非下采样轮廓波变换对输入进行处理后，得到轮廓波变换系数，例如一级分解后的式(6.66)，其个数为 $K+1$ 个特征图，每幅特征图的大小与输入一致，本质上，这一步也可视为深度卷积神经网络中的卷积操作，将非下采样轮廓波变换后得到的特征图，再接池化、非线性和批量归一化等操作，换言之，深度卷积神经网络中第一卷积流(即卷积、池化、非线性和批量归一化等操作)没有可优化的参数，将第一卷积流后的输出(也可视为输入的特征)再放入后续“卷积网络”中，实现模型参数的优化学习。这样处理的合理性在于：由于第一卷积流获取输入的浅层特征，如边缘、纹理、角等，可利用已有的、且符合初级视觉皮层的变换(例如 Gobor 变换、轮廓波变换等)所替代，尽可能地减少深度卷积网络模型所学习的参数量，间接增加训练样本集的个数，以期提升网络的泛化性能。上述过程的数学描述为：假设深度卷积神经网络的架构为：

$$\begin{cases} X^{(1)} = \mathrm{CPRN}(\boldsymbol{x}, \theta_1) \\ X^{(2)} = \mathrm{CPRN}(X^{(1)}, \theta_2) \\ \vdots \\ X^{(L)} = \mathrm{CPRN}(X^{(L-1)}, \theta_L) \\ F^{(1)} = \mathrm{FC}(X^{(L)}, \theta_{L+1}) \\ \vdots \\ F^{(T)} = \mathrm{FC}(F^{(T-1)}, \theta_{L+T}) \\ y = \sigma(F^{(T)}, \omega) \end{cases} \tag{6.67}$$

其中 CPRN 分别代表着卷积(Convolution)、池化(Pooling)、非线性激活(ReLU)和归一化(Normalization),注意这里的 CPRN 是一种组合,即卷积流,以卷积为核心操作,其他的三个操作可以选择性地添加。另外 $\theta_i(i=1,2,\cdots,L+T)$ 为相应层级或(卷积流)模块上的超参数和参数;FC 为全连接层(T 为全连接层的层数),最后得到高层抽象特征 $F^{(T)}$ 后,设计分类器或回归器,实现输出。利用非线性轮廓波变换对式(6.67)中第一卷积流中的卷积操作进行替换,即

$$X^{(1)}=\mathrm{CPRN}(\boldsymbol{x},\theta_1)\xrightarrow{\text{替换}}\mathrm{PRN}(\mathrm{NSCT}(\boldsymbol{x},\tau),\tilde{\theta}_1)\tag{6.68}$$

其中 $\mathrm{NSCT}(\boldsymbol{x},\tau)$ 为对输入 $\boldsymbol{x}$ 进行 τ 级分解后的特征图,$\tilde{\theta}^{(1)}$ 为池化、非线性和批量归一化所对应的参数指标集。得到 $X^{(1)}$ 后,式(6.67)剩余的操作与之前一致,这里不再赘述。

6.6 深度森林

本质上,深度森林性能的提升依赖于层级间与层级内随机森林以及基学习器决策树的差异性,整个网络的架构见图 6.10,主要分为两部分,一是多分辨特性的融合,操作类似于对输入进行多尺度多方向的"滤波"处理,获取输入多样性的表达;二是针对级联后的特征进行"深度"处理,每一隐层由若干个带有差异性的随机森林构成,另外,每一个随机森林中基学习器(即决策树)的差异性是通过子"数据"(即第一部分处理完后的特征)的重采样法获得,这一部分主要为了避免网络的不稳定性或提升网络的鲁棒性,将第一部分的融合特征(即第二部分的输入)级联到每一隐层上,直至最后一个隐层的输出,类似若干个"弱分类器"的结果(这里的弱分类指随机森林),最后的输出为这若干个结果进行平均、最大化操作所对应的指标。

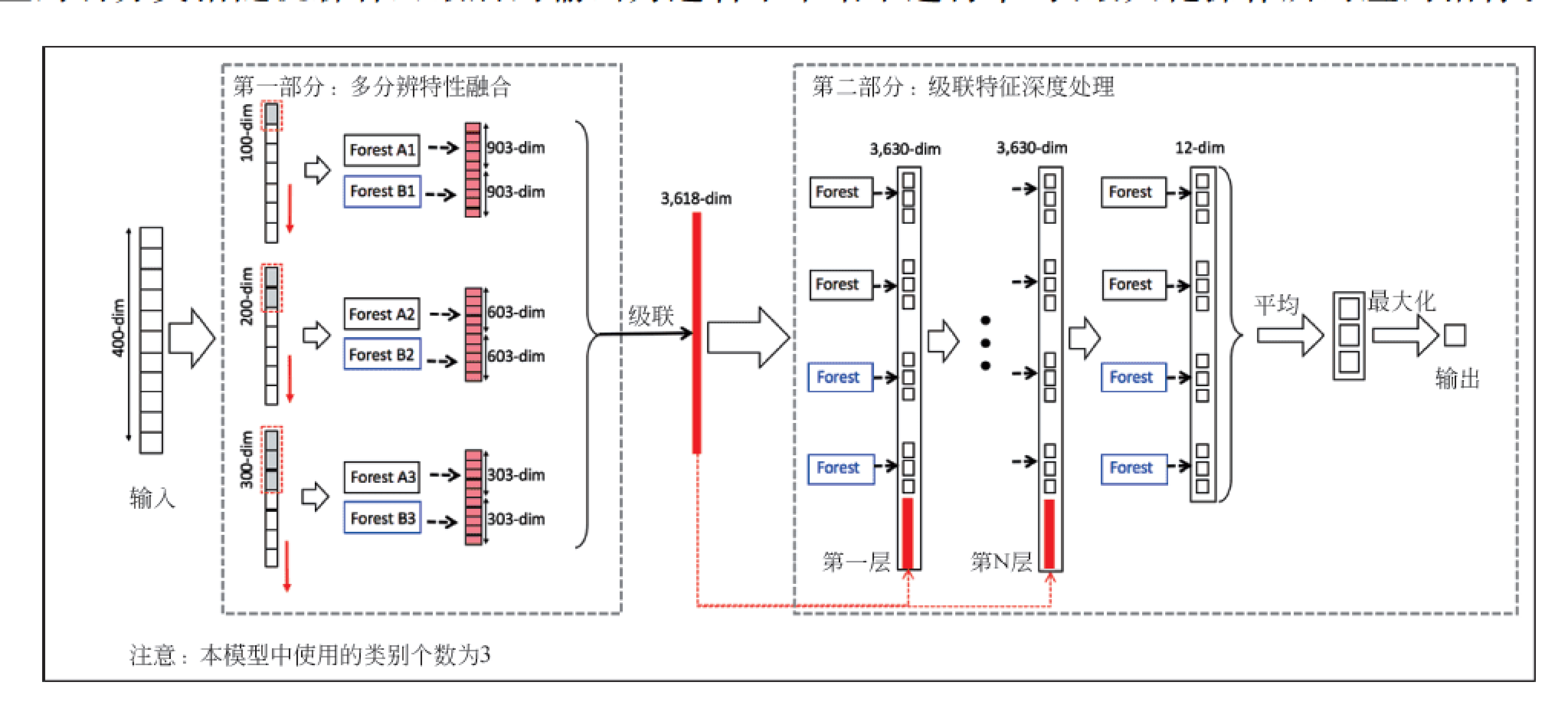


图 6.10　深度森林(多收益级联森林)的整体结构

下面我们就深度森林网络所包含的两个部分,分两节详述其具体的操作及特性。需要注意的是,本节陈述模型的应用场景是分类任务。



6.6.1　多分辨特性融合

假设分类任务中，类别个数为3；进一步，对输入 $\boldsymbol{x}\in\mathcal{R}^{400}$，窗口大小分别选择为100、200和300，可以类似理解为三个不同的“分辨特性”；对每一“分辨特性”下，利用块提取的处理方式，实现多样性的描述或局部注意。如窗口大小为100且间隔为1时，对输入进行块处理后可以得到301个子块，每一个子块为100维，即

$$\boldsymbol{x}\in\mathcal{R}^{400}\xrightarrow[\text{Stride}=1]{\text{Window}=100}\{\tilde{\boldsymbol{x}}_i\in\mathcal{R}^{100}\}_{i=1}^{301} \tag{6.69}$$

同理，对输入利用窗口大小为200和300也可进行相同处理。进一步，对每一分辨特性下所得到的每一子块，利用随机森林（由若干决策树组成）进行处理，其中随机森林的输出与分类中类别个数一致；如公式(6.69)中，对输入 $\boldsymbol{x}\in\mathcal{R}^{400}$，共计有301个子块，每一子块经过随机森林处理后，得到三维的输出，即

$$\tilde{\boldsymbol{x}}_i\in\mathcal{R}^{100}\xrightarrow{\text{Forest}}t_i\in\mathcal{R}^{3} \tag{6.70}$$

那么，将输入所对应的301个块，利用随机森林进行处理，并将处理后的结果进行堆栈级联：

$$\{\tilde{t}_i\in\mathcal{R}^{3}\}_{i=1}^{301}\xrightarrow{\text{级联}}\boldsymbol{t}=(\tilde{t}_1,\cdots,\tilde{t}_{301})\in\mathcal{R}^{903} \tag{6.71}$$

注意，为了描述差异性，这里采用了两个随机森林，并且这两个随机森林所对应的基学习器（即决策树）是不同形式的。如图6.11中，第一部分窗口大小为100时所对应的随机森林Forest A和Forest B。具体地，该部分图解如下所示。

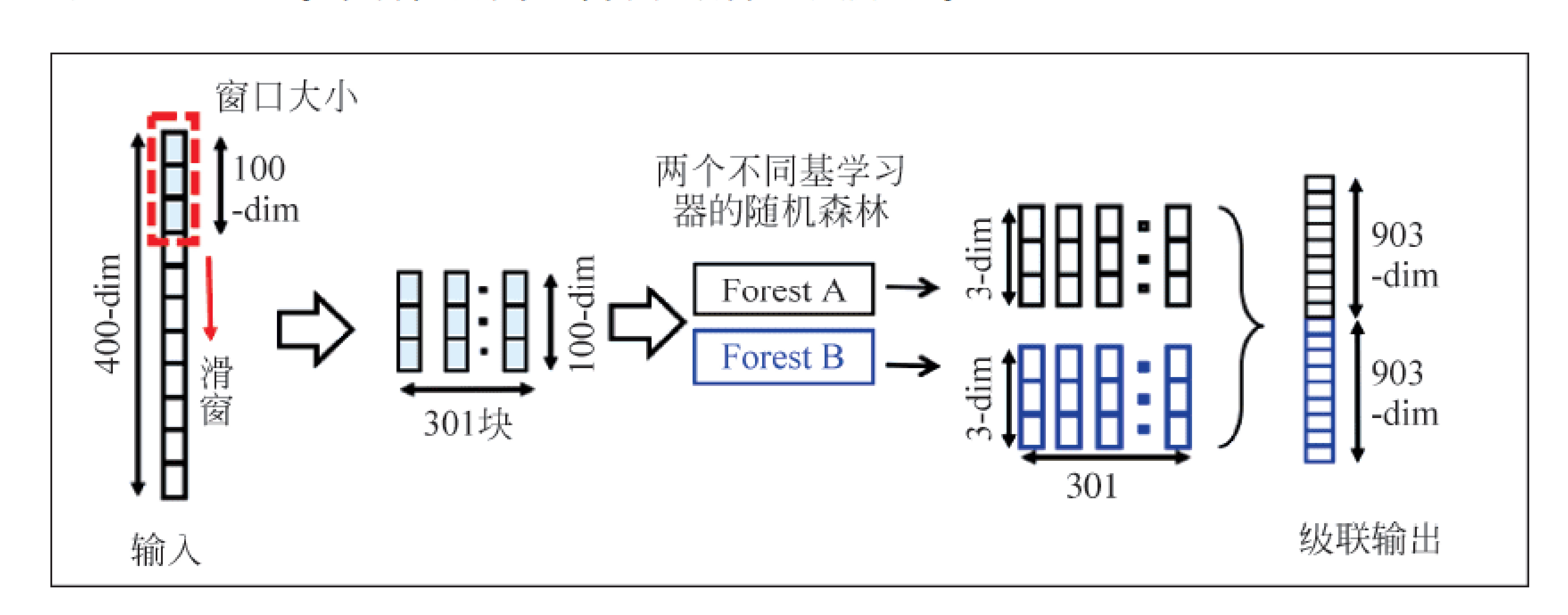


图6.11　窗口大小为100时特性融合

注意关于这两个随机森林的学习，利用有监督学习方式进行，即根据若干个输入及所对应的类标实现随机森林中参数的学习，值得指出的是每一输入经过滑窗处理后，所得到的301个块所对应的类标是一致的。第一部分，依据窗口的个数及随机森林的个数，得到多分辨特性融合后的特征为3618维。

6.6.2　级联特征深度处理

第二部分的输入为第一部分的输出，即 $\boldsymbol{t}_c\in\mathcal{R}^{3618}$，其中的角标“$c$”为不同窗口大小下特

征的级联 concatenate。为了体现"深度"的表征能力，以及层级间的差异性和多样性，这一部分的网络模型设计如图 6.12 所示。

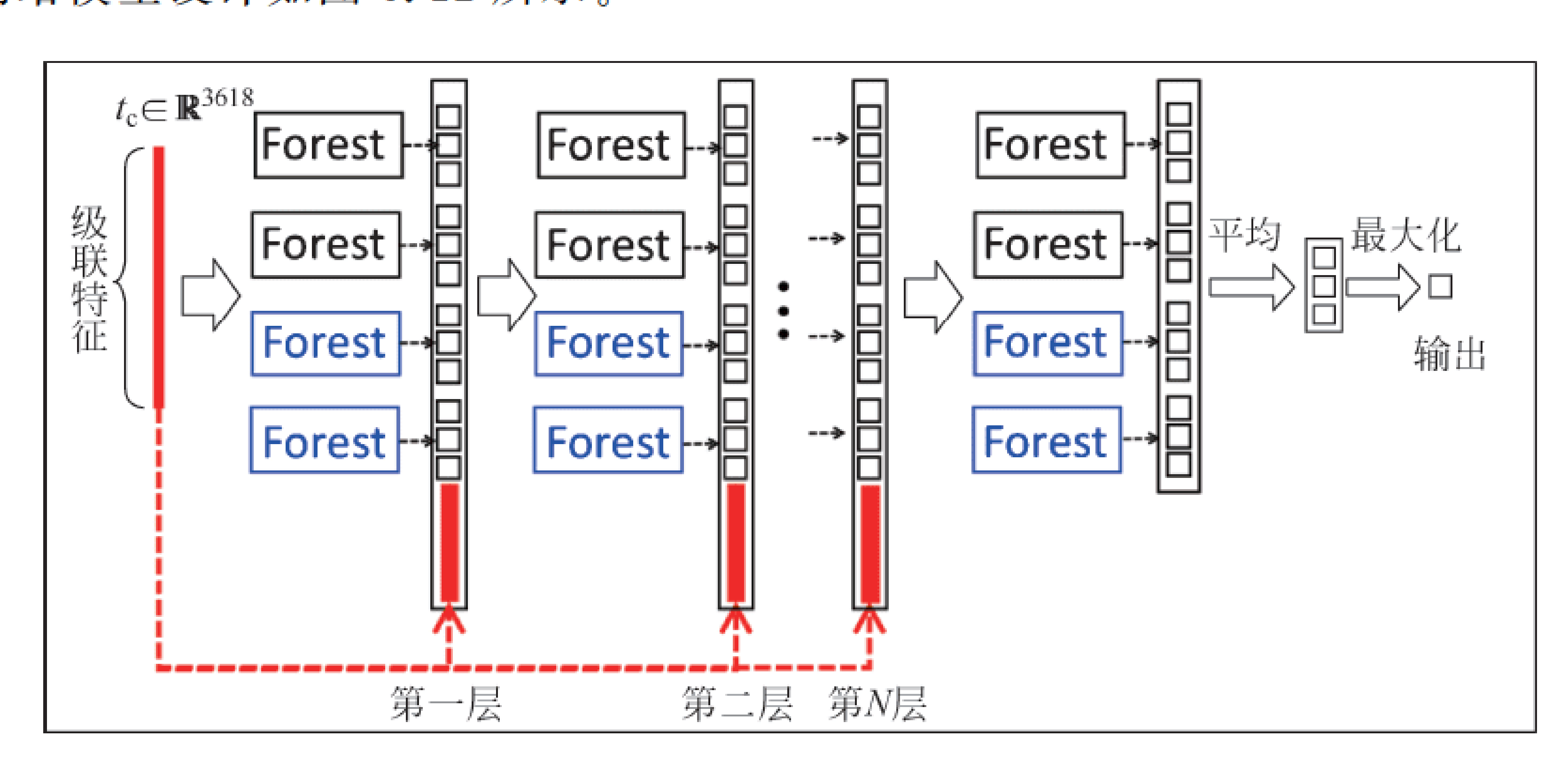


图 6.12　级联特征深度处理——分类任务

这里"深度"采用了 N 层，每一层上采用了 4 个随机森林，它们的差异性主要由基学习器(决策树)的形式和个数决定。另外，为了使得每一层上信息最大化地保留，类似残差网络中的基本模块，这里将级联特征继续以级联的方式嵌入至网络中的每一层；注意这一部分整个网络中的每一个随机森林的输出都与分类任务所对应的类标一致，即为 3。所以经过这 N 层处理后，得到的特征维数一直是 3630 维(即级联特征的维数 3618 与 4 个随机森林输出维数 12 的和)；之后在分类器设计的阶段，类似集成分类器，仍采用 4 个随机森林，对其输出进行平均和最大化的操作，得到输入 $\boldsymbol{x} \in \mathcal{R}^{400}$ 所对应的类标，即

$$\begin{cases} \boldsymbol{f}_l = P_{\text{Forest}}^l(\boldsymbol{f}_{l-1}, \boldsymbol{\theta}_l) \in \mathcal{R}^{3630} \\ \boldsymbol{f}_0 = \boldsymbol{t}_c \in \mathcal{R}^{3618} \\ \boldsymbol{\theta}_l = (\boldsymbol{\theta}_{l,1}, \boldsymbol{\theta}_{l,2}, \boldsymbol{\theta}_{l,3}, \boldsymbol{\theta}_{l,4}) \end{cases} \tag{6.72}$$

其中 $l=1,2,\cdots,N$；参数$\boldsymbol{\theta}_l$ 为第 l 层上 4 个随机森林所对应的参数；操作"P_{Forest}^l"代表第 l 层上随机森林、级联特征嵌入等处理。

进一步，对于分类器设计，则有

$$\begin{cases} \boldsymbol{f}_{N+1,k} = F_{N+1,k}(\boldsymbol{f}_N, \boldsymbol{\theta}_{N+1,k}) \in \mathcal{R}^3 \\ \bar{\boldsymbol{f}} = \dfrac{1}{4}\sum_{k=1}^{4} \boldsymbol{f}_{N+1,k} \in \mathcal{R}^3 \\ \text{label}(x) = \arg\max_{j=1,2,3} \{\bar{\boldsymbol{f}}_j\} \end{cases} \tag{6.73}$$

操作"$F_{N+1,k}$"为第 $N+1$ 层上第 k 个随机森林的输出，注意与操作"P_{Forest}^l"的区别。

值得注意的是：整个网络有没有非线性操作呢？有，类似于 Maxout 网络。另外，小规模数据的差异性以及层内和层间随机森林的差异性，能否随着层级的加深，使得对(复杂)输入的表征能力进一步提升？另外，对于每一个森林，其基学习器(即决策树)的形式与个数决定着随机森林的性能以及抗干扰水平，可以预知，个数越少，预训练时，随机森林越难于收敛

并且性能较差；但个数越多，将会影响整个网络的计算消耗。当然，该网络的确扩展了深度学习的思路，不仅仅是基于神经网络的实现。同时，它的优点也充分体现了“深度”结合机器学习中简单算法的特性：简洁高效的优化方式与模型表征能力的提升。

参考文献

[6.1] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521(7553): 436-444.

[6.2] Han S, Mao H, Dally W J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding[J]. Fiber, 2016, 56(4): 3-7.

[6.3] Peng X B, Berseth G, Michiel V D P. Terrain-adaptive locomotion skills using deep reinforcement learning[J]. ACM Transactions on Graphics, 2016, 35(4): 81.

[6.4] Peng X B, Panne M V D. Learning Locomotion Skills Using DeepRL: Does the Choice of Action Space Matter? [J]. 2016.

[6.5] Sun Z J, Xue L, Yang-Ming X U, et al. Overview of deep learning[J]. Application Research of Computers, 2012.

[6.6] Wu J, Zhang C, Xue T, et al. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling[J]. 2016.

[6.7] Mirza M, Osindero S. Conditional Generative Adversarial Nets[J]. Computer Science, 2014: 2672-2680.

[6.8] Goodfellow I J, Pougetabadie J, Mirza M, et al. Generative Adversarial Networks[J]. Advances in Neural Information Processing Systems, 2014, 3: 2672-2680.

[6.9] Gauthier J. Conditional generative adversarial nets for convolutional face generation[J]. Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester, 2014.

[6.10] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks[J]. Computer Science, 2015.

[6.11] Denton E, Chintala S, Szlam A, et al. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks[J]. Computer Science, 2015.

[6.12] Liu M Y, Tuzel O. Coupled Generative Adversarial Networks[J]. 2016.

[6.13] Chen M, Denoyer L. Multi-view Generative Adversarial Networks[J]. 2016.

[6.14] Arjovsky M, Bottou L. Towards Principled Methods for Training Generative Adversarial Networks[J]. 2017.

[6.15] Schuster M, Paliwal K K. Bidirectional recurrent neural networks[J]. IEEE Transactions on Signal Processing, 1997, 45(11): 2673-2681.

[6.16] Graves A. Supervised Sequence Labelling with Recurrent Neural Networks[M]. Springer Berlin Heidelberg, 2012.

[6.17] Graves A, Mohamed A, Hinton G. Speech Recognition with Deep Recurrent Neural Networks [J]. 2013, 38(2003): 6645-6649.

[6.18] Krueger D, Memisevic R. Regularizing RNNs by Stabilizing Activations [J]. Computer Science, 2015.

[6.19] Ji S，Vishwanathan S V N，Satish N，et al. BlackOut：Speeding up Recurrent Neural Network Language Models With Very Large Vocabularies[J]. Computer Science，2015，115(8)：2159-2168.

[6.20] Choi E，Bahadori M T，Schuetz A，et al. Doctor AI：Predicting Clinical Events via Recurrent Neural Networks[J]. Computer Science，2015.

[6.21] Talathi S S，Vartak A. Improving Performance of Recurrent Neural Network with ReLU Nonlinearity[J]. Computer Science，2015.

[6.22] Karpathy A，Johnson J，Li F F. Visualizing and Understanding Recurrent Networks[J]. 2015.

[6.23] Hidasi B，Karatzoglou A，Baltrunas L，et al. Session-based Recommendations with Recurrent Neural Networks[J]. Computer Science，2015.

[6.24] Bashivan P，Rish I，Yeasin M，et al. Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks[J]. Computer Science，2015.

[6.25] Bell S，Zitnick C L，Bala K，et al. Inside-Outside Net：Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks[J]. 2015：2874-2883.

[6.26] Collins J，Sohldickstein J，Sussillo D. Capacity and Trainability in Recurrent Neural Networks [J]. 2016.

[6.27] He K，Zhang X，Ren S，et al. Deep Residual Learning for Image Recognition[C]. IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society，2016：770-778.

[6.28] Jou B，Chang S F. Deep Cross Residual Learning for Multitask Visual Recognition[C]. ACM on Multimedia Conference. ACM，2016：998-1007.

[6.29] He K，Zhang X，Ren S，et al. Identity Mappings in Deep Residual Networks[J]. 2016.

[6.30] Ishikawa Y，Washiya K，Aoki K，et al. Brain tumor classification of microscopy images using deep residual learning[C]. SPIE BioPhotonics Australasia. 2016：100132Y.

[6.31] Han Y S，Yoo J，Ye J C. Deep Residual Learning for Compressed Sensing CT Reconstruction via Persistent Homology Analysis[J]. 2016.

[6.32] Bae W，Yoo J，Ye J C. Beyond Deep Residual Learning for Image Restoration：Persistent Homology-Guided Manifold Simplification[J]. 2016.

[6.33] Yang W，Feng J，Yang J，et al. Deep Edge Guided Recurrent Residual Learning for Image Super-Resolution[J]. 2016.

[6.34] Shah A，Kadam E，Shah H，et al. Deep Residual Networks with Exponential Linear Unit[C]. International Symposium on Computer Vision and the Internet. ACM，2016：59-65.

[6.35] Boyd S，Parikh N，Chu E，et al. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers[J]. Foundations & Trends® in Machine Learning，2010，3(1)：1-122.

[6.36] Y Yang，J Sun，H Li，et al. Deep ADMM-Net for Compressive Sensing MRI，2016.

[6.37] Larsson G，Maire M，Shakhnarovich G. FractalNet：Ultra-Deep Neural Networks without Residuals[J]. 2017.

[6.38] Mnih V，Kavukcuoglu K，Silver D，et al. Playing Atari with Deep Reinforcement Learning[J]. Computer Science，2013.

[6.39] Mnih V，Kavukcuoglu K，Silver D，et al. Human-level control through deep reinforcement learning. [J]. Nature，2015，518(7540)：529-533.



[6.40] Narasimhan K, Kulkarni T, Barzilay R. Language Understanding for Text-based Games Using Deep Reinforcement Learning[J]. Computer Science, 2015, 40(4): 1-5.

[6.41] Hasselt H V, Guez A, Silver D. Deep Reinforcement Learning with Double Q-learning[J]. Computer Science, 2015.

[6.42] Wang Z, Schaul T, Hessel M, et al. Dueling Network Architectures for Deep Reinforcement Learning[J]. Computer Science, 2016.

[6.43] Zoph B, Le Q V. Neural Architecture Search with Reinforcement Learning[J]. 2016.

[6.44] Jaques N, Gu S, Turner R E, et al. Tuning Recurrent Neural Networks with Reinforcement Learning[J]. 2016.

[6.45] Kulkarni T D, Saeedi A, Gautam S, et al. Deep Successor Reinforcement Learning[J]. 2016.

[6.46] Baker B, Gupta O, Naik N, et al. Designing Neural Network Architectures using Reinforcement Learning[J]. 2016.

[6.47] Kulkarni T D, Narasimhan K R, Saeedi A, et al. Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation[J]. 2016.

[6.48] Zhou Z H, Feng J. Deep Forest: Towards An Alternative to Deep Neural Networks[J]. 2017.

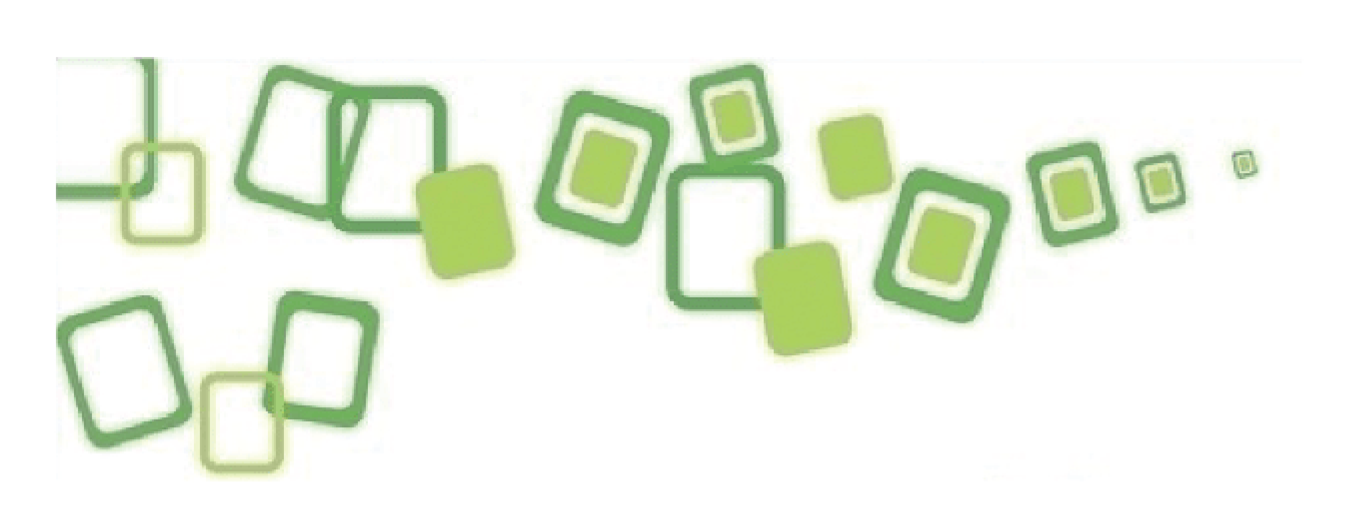
[6.49] Wiering M A, Schutten M, Millea A, et al. Deep Support Vector Machines for Regression Problems[C]. The Workshop on Advances in Regularization, Optimization, Kernel Methods, and Support Vector Machines: Theory and Applications. 2013: 141-148.

[6.50] Tang J, Deng C, Huang G B. Extreme Learning Machine for Multilayer Perceptron[J]. IEEE Transactions on Neural Networks & Learning Systems, 2016, 27(4): 809.

[6.51] Ding S, Zhang N, Xu X, et al. Deep Extreme Learning Machine and Its Application in EEG Classification[J]. Mathematical Problems in Engineering, 2015, 2015(1): 1-11.

[6.52] Yu W, Zhuang F, He Q, et al. Learning deep representations via extreme learning machines[J]. Neurocomputing, 2015, 149: 308-315.

第7章



深度生成网络

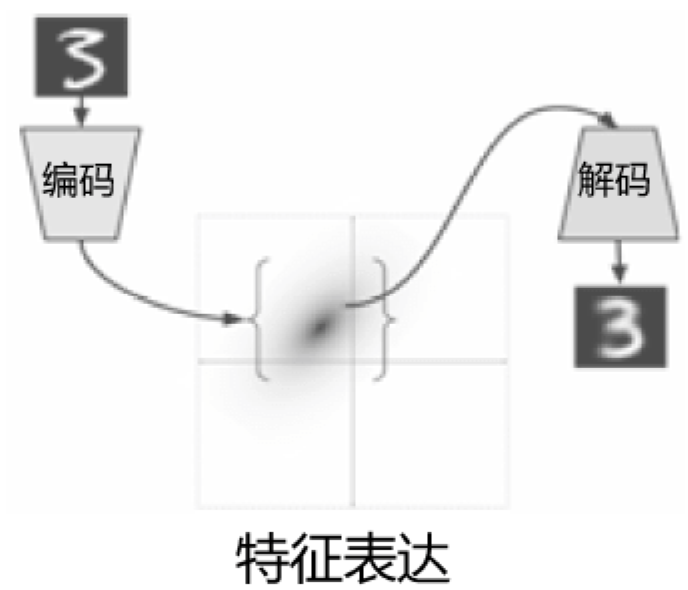
CHAPTER 7

深度生成网络——生成样本、数据缺失填补

■ 生成式对抗网络——数据空间

■ 变分自编码网络——特征空间







7.1 生成式对抗网络的基本原理

生成式对抗网络，英文名称为 Generative Adversarial Nets，2014 年 6 月 Ian Goodfellow 等学者提出，它是一种生成模型，核心思想是从训练样本中学习所对应的概率分布，以期根据概率分布函数获取更多的“生成”样本来实现数据的扩张。另外，它包括两个子网络模型，一个是生成模型（使得生成的“伪”图像尽可能与“自然”图像的分布一致），一个是判别模型（在生成的“伪”图像与“自然”图像之间作出正确判断，即二分类器），实现整个网络训练的方法便是让这两个网络相互竞争，最终生成模型通过学习“自然”数据的本质特性，从而刻画出“自然”样本的分布概型，生成与“自然”样本相似的新数据。下面为了进一步理解和分析该网络，我们主要从网络模型的动机和数学物理描述两个方面进行陈述。

备注：生成式对抗网络与“对抗”样本没有关系，对抗样本指对数据进行统计处理，如加入随机噪声等，得到新数据，使得模型对新数据并不能很好地进行预测，当然 Ian Goodfellow 很好地解释了原因：因为新数据噪声部分与网络模型进行前向计算后（即与权值矩阵相乘，与偏置相加），其值较大影响了最后输出的决策。

7.1.1 网络模型的动机

1. 生物或类脑机理——人类行为

生成式对抗网络的启发主要源于博弈论中的二人零和博弈，即指参与博弈的双方，在严格竞争下，一方的收益必然意味着另一方的损失，博弈双方的收益和损失相加总和永远为“零”，双方不存在合作。对于非合作、纯竞争型博弈，例如两个人打乒乓球，一个人赢则意味着另一人输；抽象后的博弈问题为：已知参与者集合（双方），策略集合（乒乓球技术水平）和盈利集合（胜负），能否找到一个理论上的平衡点，即对参与双方来说都最合理、最优的具体策略？冯·诺依曼已经从数学上证明，对二人零和博弈问题，可以通过一定的线性运算操作（即竞争双方以概率分布的形式随机使用某类最优策略中的各个策略），找到一个最小最大的平衡点，这个著名的最小最大定理的思想是抱最好的希望，做最坏的打算。

对于拓扑结构不完整的场景，生物视觉皮层（腹侧视觉通路）对其的识别随着层级（即从外侧膝状体，到初级视觉皮层，再到中级视觉皮层，最后到颞叶皮层和决策层等）的不断抽象而呈现“盲区”，即无法判断场景是什么，基于此构建的深度神经网络模型难以实现（输入或场景的）拒识。近几年，关于大脑和生物视觉皮层的最新研究发现，当每一层级获取关于场景的拓扑结构信息后，对该信息具有修正或（先验）匹对等生物学功能的皮层区（记为 Recovery Functional Area）可以进一步修复或完善该（拓扑结构不完整的）场景的拓扑对应性。若场景拓扑结构完整，则不断增强 Recovery Functional Area 的记忆存储，并对腹侧视觉通路进行 Positive 刺激；若场景拓扑结构不完整，则弱化腹侧视觉通路并给出 Negative 刺激，同时利用 Recovery Functional Area 对其（即腹侧视觉通路获取的层级信息）进行完

善，如图 7.1 所示。

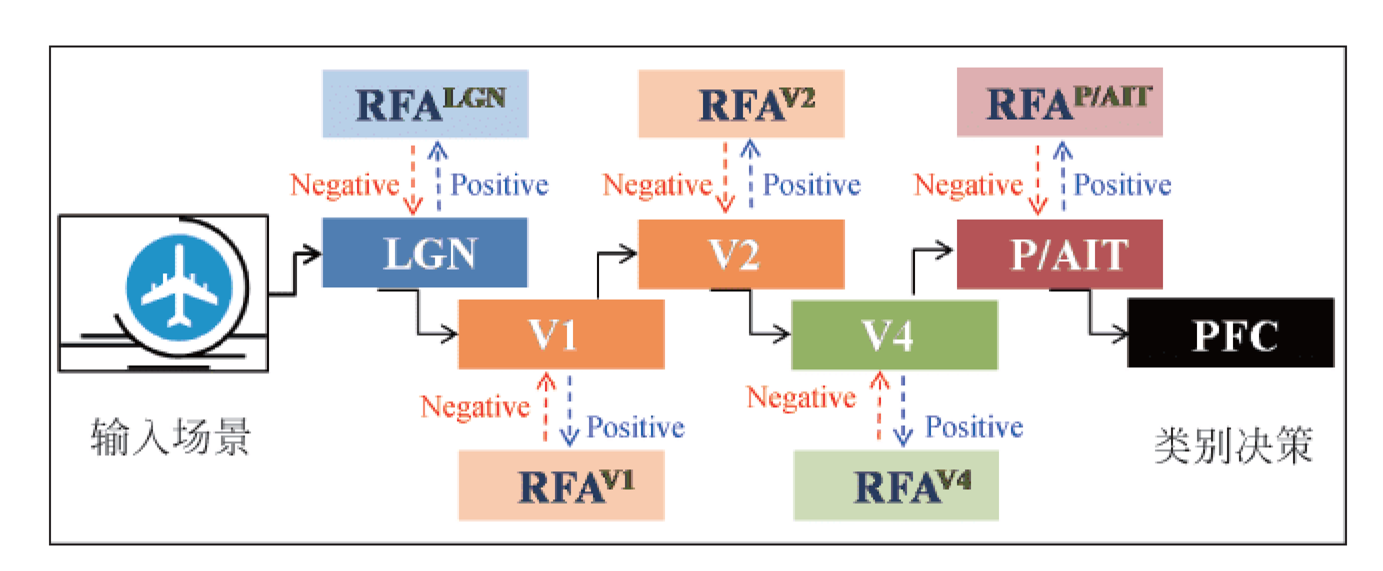


图 7.1　腹侧视觉通路与恢复功能区零和博弈

注意符号 RFA，即恢复功能区；可以认为“显”网络为腹侧视觉通路，“隐”网络为相对应层上的恢复功能区，这两条网络相互博弈（即输入拓扑结构完整时，所有皮层上对应恢复功能区上为 Positive 刺激；如输入为不完整的拓扑结构，则对应 Negative 刺激），对输入场景实现高性能的类别决策。

2. 数据扩张

众所周知，深度学习的主要驱动力为可利用的数据量（即输入与输出），数据量越充分，训练得到的模型泛化能力（测试性能）越好。但在实际应用中，带有标记的数据很少，且代价昂贵；除了常用的统计扩张数据方式（例如裁剪、滑块和旋转角度、多分辨非下采样处理、加入服从不同分布下的随机噪声等，本质上，得到的样本可视为对抗样本且以多分辨特性、容许旋转不变性和鲁棒性等融入至模型内，但模型的预测（外插）能力受限于这种扩充方式）外，生成式对抗网络也可以无监督学习的方式实现数据的扩张，如图 7.2 所示。

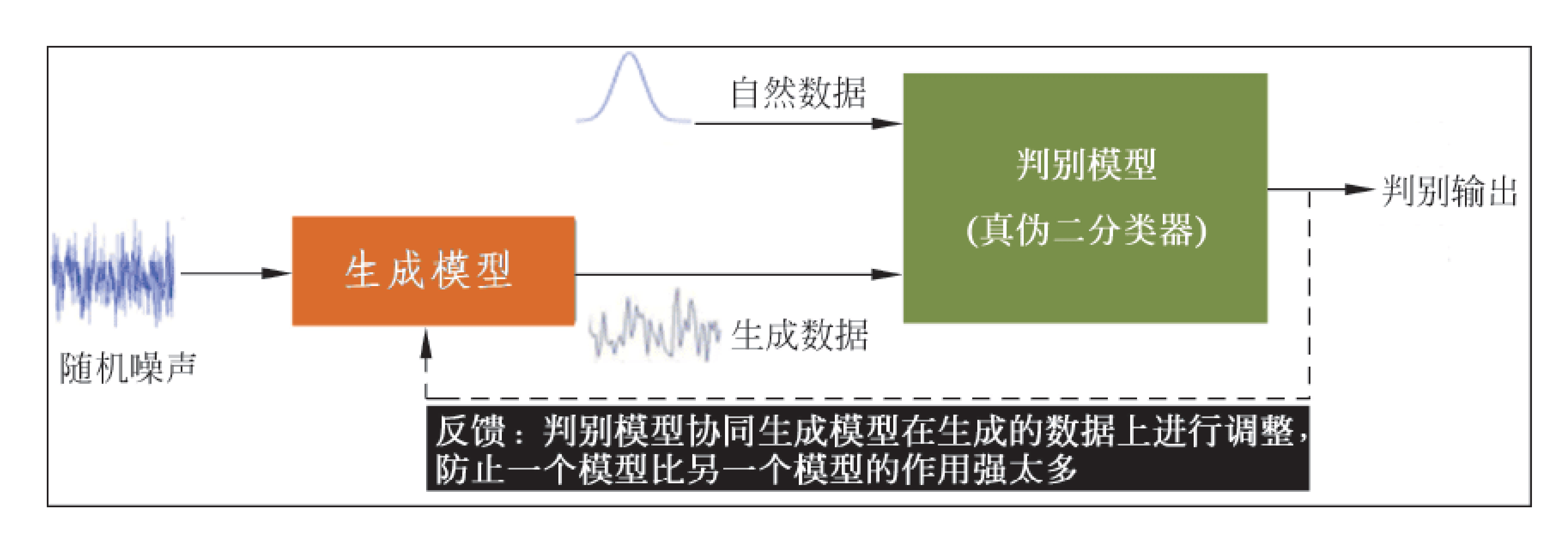
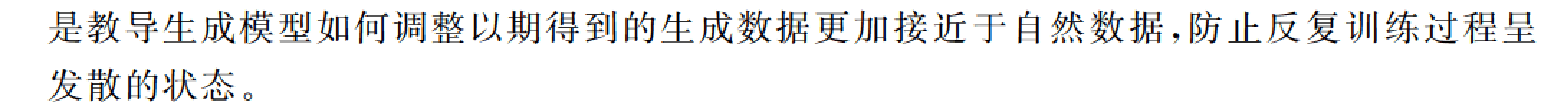


图 7.2　基于生成式对抗网络的数据扩张

当生成模型与判别模型交替优化学习，最终达到零和博弈（纳什均衡）时，注意生成模型的性能（即伪造（生成）数据的拓扑（几何）结构特性）取决于训练（自然）数据的量，实际应用中，若生成模型的参数远小于训练数据的量，则可以有效地内化数据的分布特性，从而使得生成的数据“接近”这些自然数据。另外网络中两个模型的设计和交替优化算法的设置也十分重要，注意生成式对抗网络的主要目的是优化生成模型，但判别模型在这里所起到的作用



是教导生成模型如何调整以期得到的生成数据更加接近于自然数据，防止反复训练过程呈发散的状态。

7.1.2　网络模型的数学物理描述

下面基于图7.2给出生成式对抗网络的数学原理及物理解释，首先，符号描述为：随机噪声 $\boldsymbol{z}\in\mathbb{R}^m$，自然数据 $\boldsymbol{x}\in\mathbb{R}^n$，生成数据为$\tilde{\boldsymbol{x}}\in\mathbb{R}^n$；由于判别模型为二分类器，所以 $\boldsymbol{y}\in[0,1]^2$。接下来，从以下四个方面详述：

1. 数据

$$\{(\boldsymbol{x}^{(t)},\boldsymbol{z}^{(t)}),\boldsymbol{y}^{(t)}\}_{t=1}^{T} \tag{7.1}$$

对于第 t 个数据对$(\boldsymbol{x}^{(t)},\boldsymbol{z}^{(t)})$，所对应的输出 $\boldsymbol{y}^{(t)}$ 为$[1,0]$，表示将自然数据判断为真的概率为1，将生成数据判断为真的概率为0；或$[0,1]$表示将自然数据判断为伪的概率为0，将生成数据判断为伪的概率为1。通常，深度学习平台 Tensorflow 采用的第一种取法。需要注意噪声数据的量不一定与自然数据的量一致，这里为了方便设置为一致。

2. 模型

$$\begin{cases} G:\tilde{\boldsymbol{x}}=g(\boldsymbol{z},\theta^G)\in\mathbb{R}^n \\ D:\begin{cases} \text{Feature Learning}:\begin{cases}\boldsymbol{X}=D^F(\boldsymbol{x},\theta^F)\\ \tilde{\boldsymbol{X}}=D^F(\tilde{\boldsymbol{x}},\theta^F)\end{cases} \\ \text{Classifier Design}:\boldsymbol{y}=\begin{pmatrix}P(L(\boldsymbol{x})=\text{real}\mid\boldsymbol{X},\theta^C)\\ P(L(\tilde{\boldsymbol{x}})=\text{real}\mid\tilde{\boldsymbol{X}},\theta^C)\end{pmatrix}\in\mathbb{R}^2\end{cases}\end{cases} \tag{7.2}$$

其中的 G 表示 Generator，即生成模型或称生成器，待优化参数为 θ^G，需进一步量化非线性映射函数 $g(\cdot)$；另外 D 表示 Discriminator，即判别模型或判别器，分为两个阶段，一个是特征学习，待优化的参数为 θ^F，另一个是分类器设计，待优化的参数为 θ^C；同样需量化映射 $D^F(\cdot)$和 $P(\cdot)$这两个过程。值得注意的是：数据量要远大于生成模型的参数量，即 $T\gg\text{Num}(\theta^G)$，才可以保证网络得到零和博弈解，另外 $L(\boldsymbol{x})$为输入 $\boldsymbol{x}$ 所对应的真伪性。

3. 优化目标函数

通常，也可以将模型(7.2)中的判别模型部分写为：

$$\boldsymbol{y}=\begin{pmatrix}D(\boldsymbol{x})\\ D(\tilde{\boldsymbol{x}})\end{pmatrix}=\begin{pmatrix}D(\boldsymbol{x})\\ D(G(\boldsymbol{z}))\end{pmatrix}\in\mathbb{R}^2 \tag{7.3}$$

其中 $D(\boldsymbol{x})\in[0,1]$将 $\boldsymbol{x}$ 判断为真样本(即自然数据)的概率。在固定生成模型时所对应判别器的损失函数为：

$$\begin{cases}\min\limits_{\theta^D}\left\{-\left[\sum\limits_{x\sim P(x)}\log(D(\boldsymbol{x},\theta^D))+\sum\limits_{\tilde{x}\sim P(\tilde{x})}\log(1-D(\tilde{\boldsymbol{x}},\theta^D))\right]\right\}\\ \theta^D=(\theta^F,\theta^C)\end{cases} \tag{7.4}$$



这里 $\boldsymbol{x}\sim P(\boldsymbol{x})$ 为服从自然数据分布 $P(\boldsymbol{x})$ 下的采样，即式(7.1)中的自然数据集，对应着 $\tilde{\boldsymbol{x}}\sim P(\tilde{\boldsymbol{x}})$ 为服从生成分布概型 $P(\tilde{\boldsymbol{x}})$ 下的采样，即式(7.1)中的生成数据集。公式(7.4)中 $-\log(D(\boldsymbol{x}))$ 的物理解释为将 $\boldsymbol{x}$ 判断为真(自然数据)的不确定性越小越好(不确定性越小则意味着确定性越高)，其最佳状态为0，即 $D(\boldsymbol{x})=1$；另外 $\log(1-D(\tilde{\boldsymbol{x}}))$ 的物理解释为将 $\tilde{\boldsymbol{x}}$ 判断为伪(生成数据)的不确定性越小越好，即 $(1-D(\tilde{\boldsymbol{x}}))$ 为将 $\tilde{\boldsymbol{x}}$ 判断为伪的概率要越大越好，此意味着 $D(\tilde{\boldsymbol{x}})$ 为将 $\tilde{\boldsymbol{x}}$ 判断为真的概率越小越好；将所有采样样本的不确定性(也称信息量)进行求和，便得到熵的概念。简言之，判别模型的设计要求为：将自然数据判断为真的概率要高，将生成数据判断为伪的概率要高。

另外，对生成模型的要求是：在判别模型的固定时，生成数据的分布特性尽最大可能与自然数据的一致，即在 $P(\tilde{\boldsymbol{x}})$ 尽可能与 $P(\boldsymbol{x})$ 一致的情形下，最大化如下目标函数：

$$\max_{\theta^G}\sum_{\tilde{\boldsymbol{x}}\sim P(\tilde{\boldsymbol{x}})}\log(D(\tilde{\boldsymbol{x}}))=\sum_{\tilde{\boldsymbol{x}}\sim P(\boldsymbol{x})}\log(D(\tilde{\boldsymbol{x}})) \tag{7.5}$$

对应着，将 $\boldsymbol{x}=G(\boldsymbol{z})$ 代入便有：

$$\max_{\theta^G}\sum_{\boldsymbol{z}\sim P(\boldsymbol{z})}\log(D(G(\boldsymbol{z},\theta^G)))\xrightarrow{\text{衡量}}d(P(\tilde{\boldsymbol{x}}),P(\boldsymbol{x})) \tag{7.6}$$

即在 $\boldsymbol{z}\sim P(\boldsymbol{z})$ 的条件下，所有关于 $\boldsymbol{z}$ 的 $\log(D(G(\boldsymbol{z})))$ 的和越大，意味着：

$$(D(G(\boldsymbol{z}))\sim P(\tilde{\boldsymbol{x}}))\longrightarrow d(G(\boldsymbol{z}),\boldsymbol{x})\longrightarrow(D(G(\boldsymbol{z}))\sim P(\boldsymbol{x})) \tag{7.7}$$

生成数据与自然数据之间的差距 $d(G(\boldsymbol{z}),\boldsymbol{x})$ 越小，即最为理想的状态是：关于所有的 $\boldsymbol{z}$，若都有 $\log(D(G(\boldsymbol{z})))=0$，则意味着 $D(G(\boldsymbol{z}))=1$，即将生成数据判别为自然数据(注意这是在生成模型阶段的要求)，即 $D(G(\boldsymbol{z}))$ 服从于自然数据的分布概型 $P(\tilde{\boldsymbol{x}})$，最终达到这两个分布概型 $d(P(\tilde{\boldsymbol{x}}),P(\boldsymbol{x}))$ 尽可能接近。

最后，依据式(7.1)中的数据，结合式(7.4)的损失函数，得到基于判别模型的优化目标函数为：

$$\min_{\theta^D}J(\theta^D)=\left\{-\frac{1}{T}\left[\begin{array}{l}\sum_{t=1}^{T}\delta(y^{(t)}(1)=\text{real})\cdot\log(D(\boldsymbol{x}^{(t)}))+\\ \sum_{t=1}^{T}\delta(y^{(t)}(2)=\text{fake})\log(1-D(\tilde{\boldsymbol{x}}^{(t)}))\end{array}\right]\right\} \tag{7.8}$$

通常由于自然数据与生成数据分别对应着真伪类标，所以式子中蕴含：

$$\begin{cases}\delta(y^{(t)}(1)=\text{real})=1\\ \delta(y^{(t)}(2)=\text{fake})=1\end{cases} \tag{7.9}$$

其中 $\delta(\cdot)$ 为狄利克雷函数。

在优化目标公式(7.8)的基础上，然后融入生成模型的要求，得到最后优化目标函数：

$$\min_{\theta^D}\min_{\theta^G}J(\theta^D,\theta^G)=\left\{-\frac{1}{T}\left[\begin{array}{l}\sum_{t=1}^{T}\log(D(\boldsymbol{x}^{(t)},\theta^D))+\\ \sum_{t=1}^{T}\log(1-D(G(\boldsymbol{z}^{(t)},\theta^G),\theta^D))\end{array}\right]\right\} \tag{7.10}$$

注意：

$$\max_{\theta^G} \sum_{z\sim P(z)} \log(D(G(z,\theta^G))) \rightleftarrows \min_{\theta^G} \sum_{z\sim P(z)} \log(1-D(G(z,\theta^G))) \tag{7.11}$$

4. 求解

利用梯度下降方法进行参数(θ^G,θ^D)交替优化，与之前第 4 章深度堆栈网络中深度置信网络的求解类似，这里不再赘述。

7.2　深度卷积对抗生成网络

本节我们将给出基于生成式对抗网络框架的一种经典模型，即深度卷积神经网络架构下的生成式对抗网络，并分析该模型在实践过程中，关于数据、优化算法和参数设置等带来的优缺点，以及所出现现象的物理解释。另外，给出该模型下的典型应用描述。

7.2.1　网络模型的基本结构

生成式对抗网络是一个整体的框架，其中生成模型和判别模型的设计可以采纳各种深度神经网络，与传统的深度学习只包含一个网络(端到端设计，反向传播计算参数更新)相比，生成式对抗网络包含两个网络，类似对偶学习(例如翻译系统，如 A 讲中文，B 讲英文，当 A 说了一段中文后，翻译系统将其翻译为英文，B 根据自己的经验与知识对翻译的内容做修正、语法更改，并反馈给翻译系统，使得系统得到学习；进一步根据 B 修正的英文，翻译系统再将其翻译为中文，然后 A 根据之前的这段话，以及自己的经验与知识，对翻译的中文做进一步的修正，如此交替学习，直至系统稳定，注意对偶学习不同于之前的自编码网络)，进行交替优化，达到零和博弈的状态。

下面首先给出深度卷积对抗生成网络的结构，如图 7.3 所示，注意该网络的优化学习范式为无监督学习。

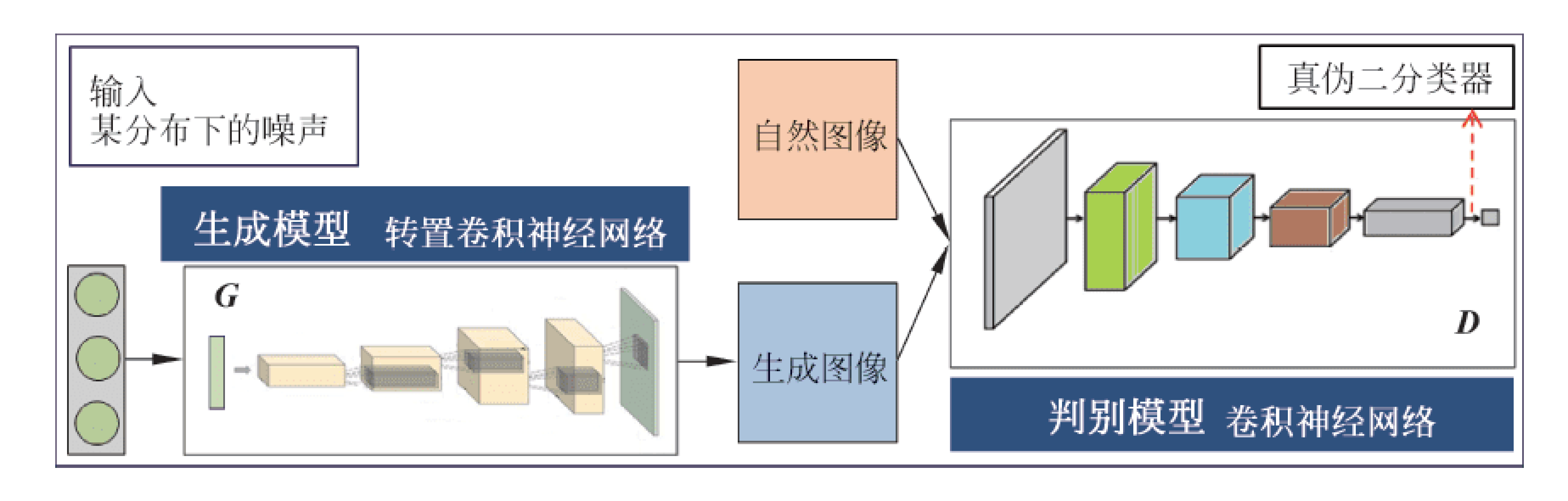


图 7.3　深度卷积对抗生成网络的结构

图 7.3 中的生成模型与判别模型可以采纳传统卷积神经网络的架构，下面通过例子来说明生成模型和判别模型。

1. 生成模型

深度卷积对抗生成网络——生成式对抗网络的结构如图 7.4 所示。

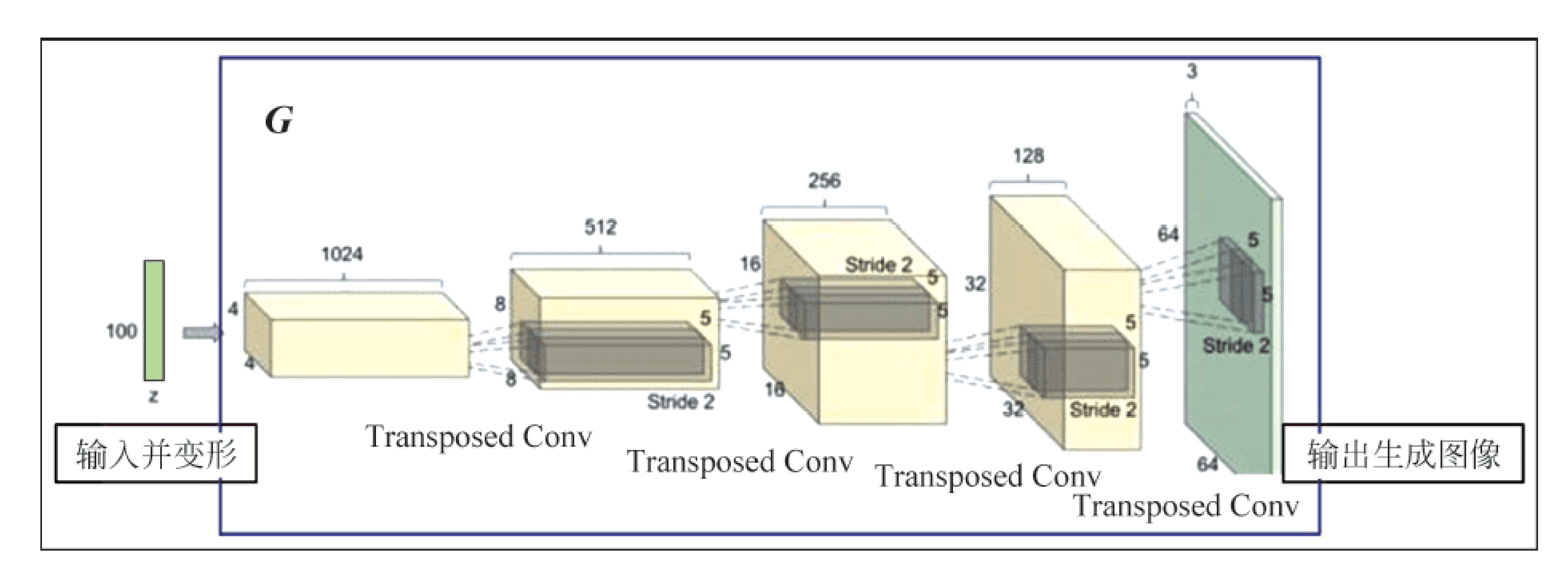


图 7.4　深度卷积对抗生成网络——生成式对抗网络的结构

假设模型的输入噪声为服从均匀分布下的随机采样，即

$$z \in \mathbb{R}^{100} \sim P(z) \tag{7.12}$$

如何从输入(噪声)利用卷积神经网络得到输出(即生成图像，其尺寸与自然图像的尺寸一致)? 注意，这里所使用的转置卷积神经网络(为了保证整个网络模型的稳定性)与之前传统的卷积神经网络有所不同，例如生成模型中的池化操作被转置(也称微步)卷积层所替代。

注意，生成模型中使用的转置卷积层操作，其中转置卷积也称微步卷积，它可以视为传统卷积操作的一种“逆向”传递过程。通常，转置卷积受“正向”卷积的参数约束，即步长 Stride 和填充方式(Zero-Padding 和 Full Padding)。这里仅给出两个小例子简要说明转置卷积与卷积的关系。更多的例子请参考：http://deeplearning.net/software/theano_versions/dev/tutorial/conv_arithmetic.html#transposed-convolution-arithmetic.

1) 无填充，步长不为 1

无填充，步长不为 1 的情况下，卷积与转置卷积的关系如图 7.5 所示。

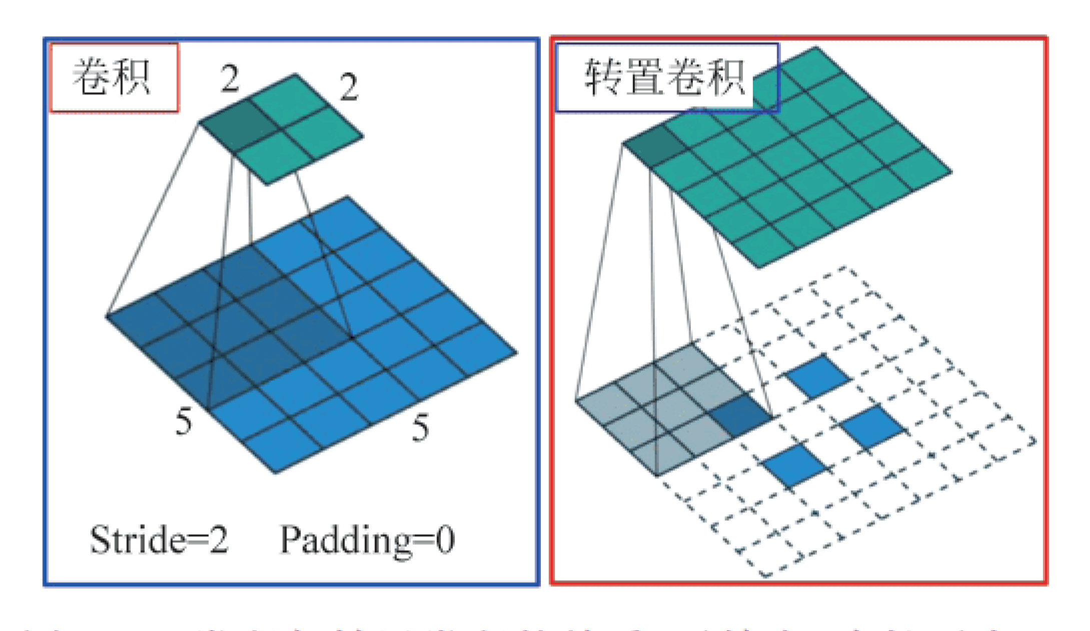
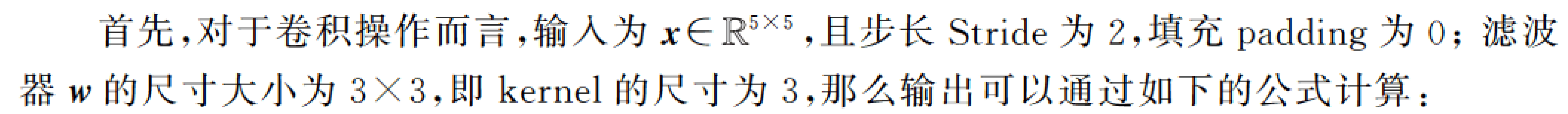


图 7.5　卷积与转置卷积的关系(无填充，步长不为 1)



首先，对于卷积操作而言，输入为 $\boldsymbol{x} \in \mathbb{R}^{5\times5}$，且步长 Stride 为 2，填充 padding 为 0；滤波器 $\boldsymbol{w}$ 的尺寸大小为 3×3，即 kernel 的尺寸为 3，那么输出可以通过如下的公式计算：

$$\text{Output} = \left\lfloor \frac{\text{Input} - \text{kernel} + 2 \cdot \text{padding}}{\text{Stride}} \right\rfloor + 1 \tag{7.13}$$

可知输出为 $\boldsymbol{y} = \text{Conv}(\boldsymbol{x}, \boldsymbol{w}, \text{'valid'}) \in \mathbb{R}^{2\times2}$，其中 $\lfloor \cdot \rfloor$ 为向下取整。

其次，对于转置卷积而言，它能够回答如何由 $\boldsymbol{y} \in \mathbb{R}^{2\times2}$ 得到 $\boldsymbol{x} \in \mathbb{R}^{5\times5}$，即卷积的"逆向"过程，这里先需要计算新的步长与填充参数，利用公式有：

$$\begin{cases} \text{Stride}^{(\text{New})} = 1 \\ \text{padding}^{(\text{New})} = \text{kernel} - 1 \end{cases} \tag{7.14}$$

注意此时的输入为 $\boldsymbol{y} \in \mathbb{R}^{2\times2}$，即新的输入尺寸，则新的输出尺寸计算如下：

$$\begin{cases} \text{Output}^{(\text{New})} = \text{Stride} \cdot (\text{Input}^{(\text{new})} - 1) + \text{kernel}^{(\text{new})} \\ \text{kernel}^{(\text{new})} = \text{kernel} \end{cases} \tag{7.15}$$

可以得到 $\boldsymbol{x} \in \mathbb{R}^{5\times5}$。

备注：式(7.14)和式(7.15)仅限于无填充且步长不为 1 的情形，但式(7.13)适用于所有"正向"卷积，即传统卷积神经网络中的操作。

2）有填充，步长不为 1

有填充，步长不为 1 的情况下的卷积与转置卷积的关系如图 7.6 所示。

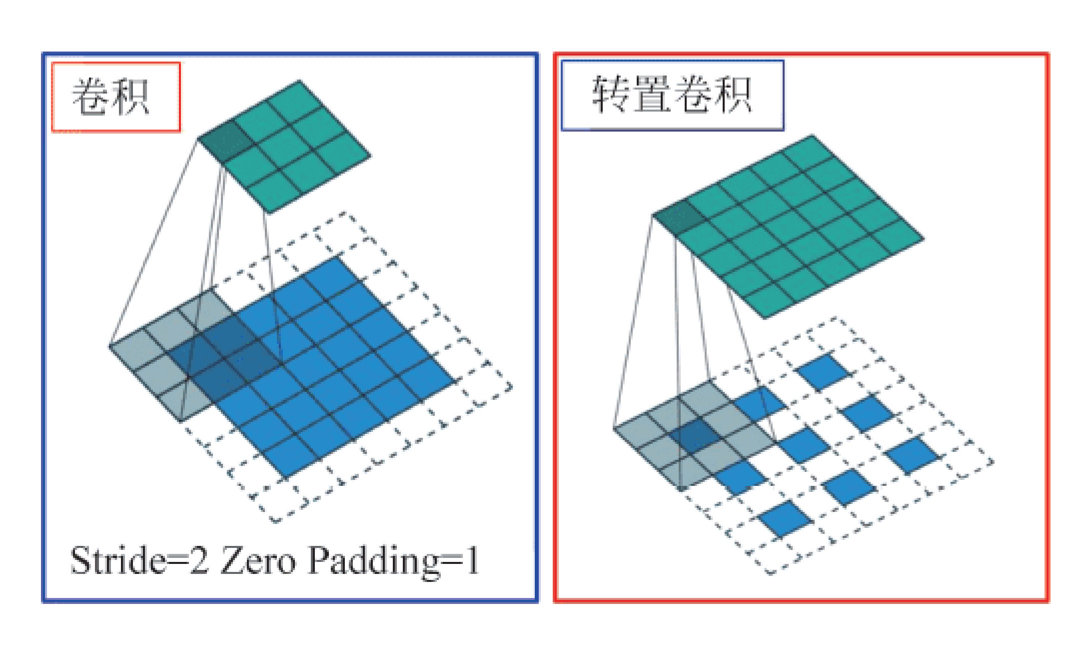


图 7.6　卷积与转置卷积的关系(有填充，步长不为 1)

同样，对于卷积操作而言，输入为 $\boldsymbol{x} \in \mathbb{R}^{5\times5}$，步长为 2，零式填充为 1，即左侧框中，输入外围的一圈"虚线框"，即每一个表示零元；那么根据式(7.13)可以计算得到输出：

$$\boldsymbol{y} = \text{Conv}(\boldsymbol{x}, \boldsymbol{w}, \text{'valid'}) \in \mathbb{R}^{3\times3} \tag{7.16}$$

其中这里的滤波器大小为 3×3。

如何"逆向"上述的卷积操作过程？先需要通过如下的公式计算新的步长和零式填充参数。

备注：注意"逆向"过程中的输入为 $\boldsymbol{y} \in \mathbb{R}^{3\times3}$。

$$\begin{cases} \text{Stride}^{(\text{New})} = 1 \\ \text{padding}^{(\text{New})} = \text{kernel} - \text{padding} - 1 \end{cases} \tag{7.17}$$

所以得到新的步长和零式填充参数分别为 1 和 1，所以右侧框中的输入外围有一圈虚线框，并且“微步”的含义是指：新的步长为 1，而之前的步长为 2，使得转置卷积的滑窗处理相比较卷积的“小”。另外，根据如下的公式计算新的输出的尺寸为：

$$\begin{cases} \text{Output}^{(\text{New})} = \text{Stride} \cdot (\text{Input}^{(\text{new})} - 1) + \text{kernel}^{(\text{new})} - 2 \cdot \text{padding} \\ \text{kernel}^{(\text{new})} = \text{kernel} \end{cases} \tag{7.18}$$

可以得到 $\boldsymbol{x} \in \mathbb{R}^{5\times 5}$。

2. 判别模型

判别模型的网络架构如图 7.7 所示。

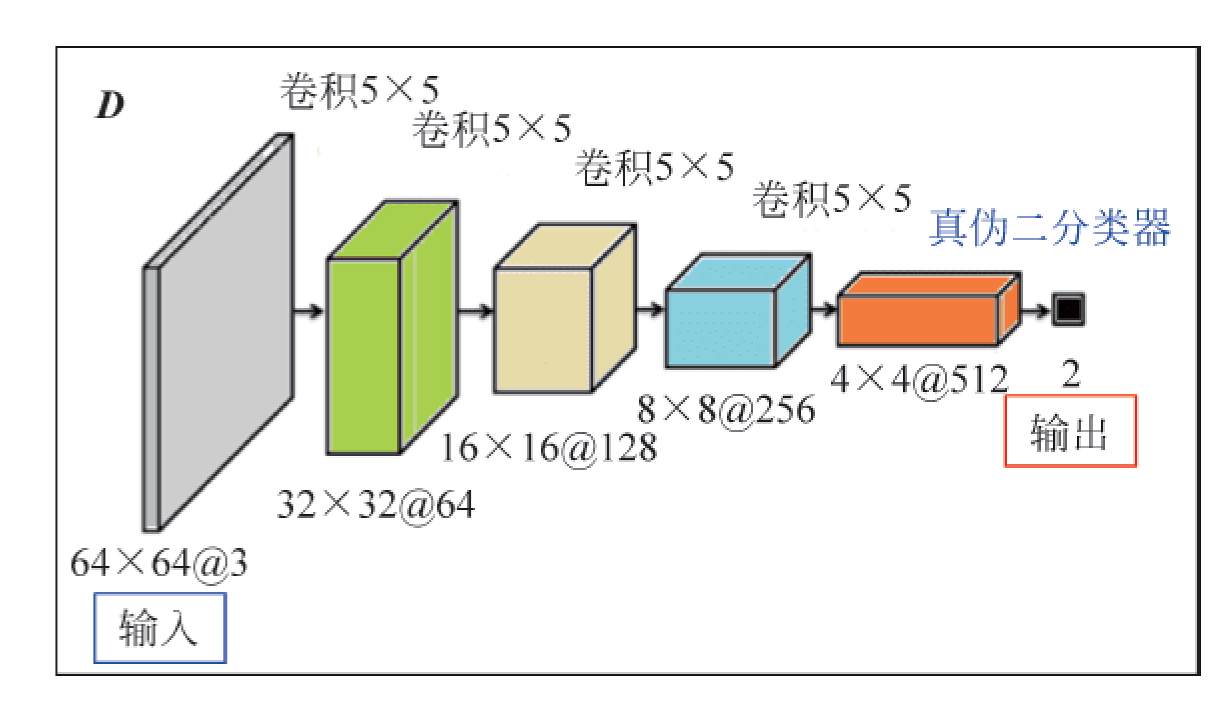


图 7.7　判别模型的网络架构

判别模型的输入为自然数据集，以及生成模型的输出(生成数据)，输出所对应的类标即为真(自然数据为 1)伪(生成数据为 0)；其中该模型可以使用传统的卷积神经网络如 LeNet、AlexNet、GoogleNet、VGGNet 等经典的网络模型，不过需要注意的是：一是真伪二分类器仍可沿用 Softmax 分类器(即退化为 Logistic 分类器)，也可以使用非线性分类器(非线性函数为 Tanh 或 Sigmoid 函数)；二是该模型中所有的池化层被卷积(融入步长)操作所替代；三是除真伪二分类器这一层外，所有隐层使用的非线性函数为修正线性单元(ReLU)的改进版 Leaky ReLU。

备注：判别模型中的卷积流，卷积、池化、非线性和批量归一化(或局部响应归一化)，除了池化层被带有步长的卷积操作所“吞噬”外，其他操作(包括特征图向量化的过程，即全连接层设计)与之前一样。深度卷积对抗网络模型更为详细的应用与深度学习平台的实现(Torch 和 Tensorflow)请参考：https://github.com/Newmu/dcgan_code.

7.2.2　网络模型的性能分析

深度卷积生成对抗网络，在具体的理论分析和应用(如分类或回归任务、超分辨等)中具有如下的特点。



1. 理论分析

众所周知，经典的生成式对抗网络 GAN 有着严格的系统理论分析与收敛性证明，即假设生成模型和判别模型都有足够的性能的条件下，如果在迭代过程中的每一步，判别模型都可以达到当下在给定生成模型时的最优值，并在这之后再更新生成模型，那么最终生成数据的概率分布函数就一定会收敛于自然数据的概率分布函数。Ian Goodfellow 等人最初假设判别模型具有无限区分能力，即不论生成数据以任意小的误差或准则接近自然数据，判别模型均可有效地识别。可问题是：若生成数据的分布函数与自然数据的分布函数接近，但其支撑集互不相交或重叠(特别当这两个分布函数是低维流型的时候，容易发生)，则生成模型所对应的优化目标函数(即詹森香农散度，它给出了生成数据与自然数据所对应分布函数之间的差异性)关于参数的偏导数退化为一常数，从而导致梯度消失，发生梯度弥散现象，换言之判别模型越好，生成模型的梯度消失越严重。

为了有效地解决这一问题，目前从理论分析上，学者提出了两个思路，一种是修正生成模型的优化目标函数，即利用沃瑟斯坦距离衡量生成数据与自然数据所对应分布函数之间的差异性，替代传统的詹森香农散度，通过最优传输定理，即生成数据的分布概型与自然数据的分布概型之间存在一个唯一的映射，记为 Wasserstein-GAN。另一种是关于判别器具有无限可分性的假设，修正为自然数据的概率分布特性具有 Lipschitz 连续且(有限阶)可微性，从而优化目标函数变为：在生成数据的分布函数与自然数据的分布函数尽可能一致的条件下(利用二者差的期望来衡量一致性)，优化判别器带有有限阶 Lipschitz 概型约束的可分能力，记为损失敏感度生成式对抗网络 Loss-Sensitive GAN，本质上，它与 Wasserstein-GAN 具有一定的相似性。

2. 应用实践

目前，对于分类任务，深度神经网络的范式主流为半监督学习，即利用无类标数据进行参数初始化，有类标数据进行网络的精调。

基于生成式对抗网络分类任务的应用主要包含两方面的贡献：一是利用生成模型进行数据扩充或利用少数有类标数据对扩充后的数据进行“类标传递”；二是利用判别模型可进行共享计算或特征学习阶段的参数初始化；注意这里的参数初始化与之前基于自编码网络的逐层学习机制有所不同，它是整个(判别模型中的特征学习)阶段的参数初始化。

当然，生成式对抗网络也并不完美，首先，优化过程存在不稳定性，很容易陷入到一个鞍点或局部极值点上；其次，模型的可解释性比较差；再次，需要提高生成式对抗网络模型的延展性，尤其在处理大规模数据的时候。相应的分析及策略如下：由于生成模型的输入为随机噪声，将会对深度卷积对抗网络的收敛产生较大的震荡，避免随机噪声的影响，实际中，常将生成模型的输入更改为自然数据的编码系数(如利用自编码网络，对输入(自然数据)进行编码后的编码系数)，这样一方面可以增加生成网络模型的拓扑(几何)结构对应性，另一方面可以提升整体网络的稳定性。另外模型的设计通常需要自然数据的个数要远大于生成

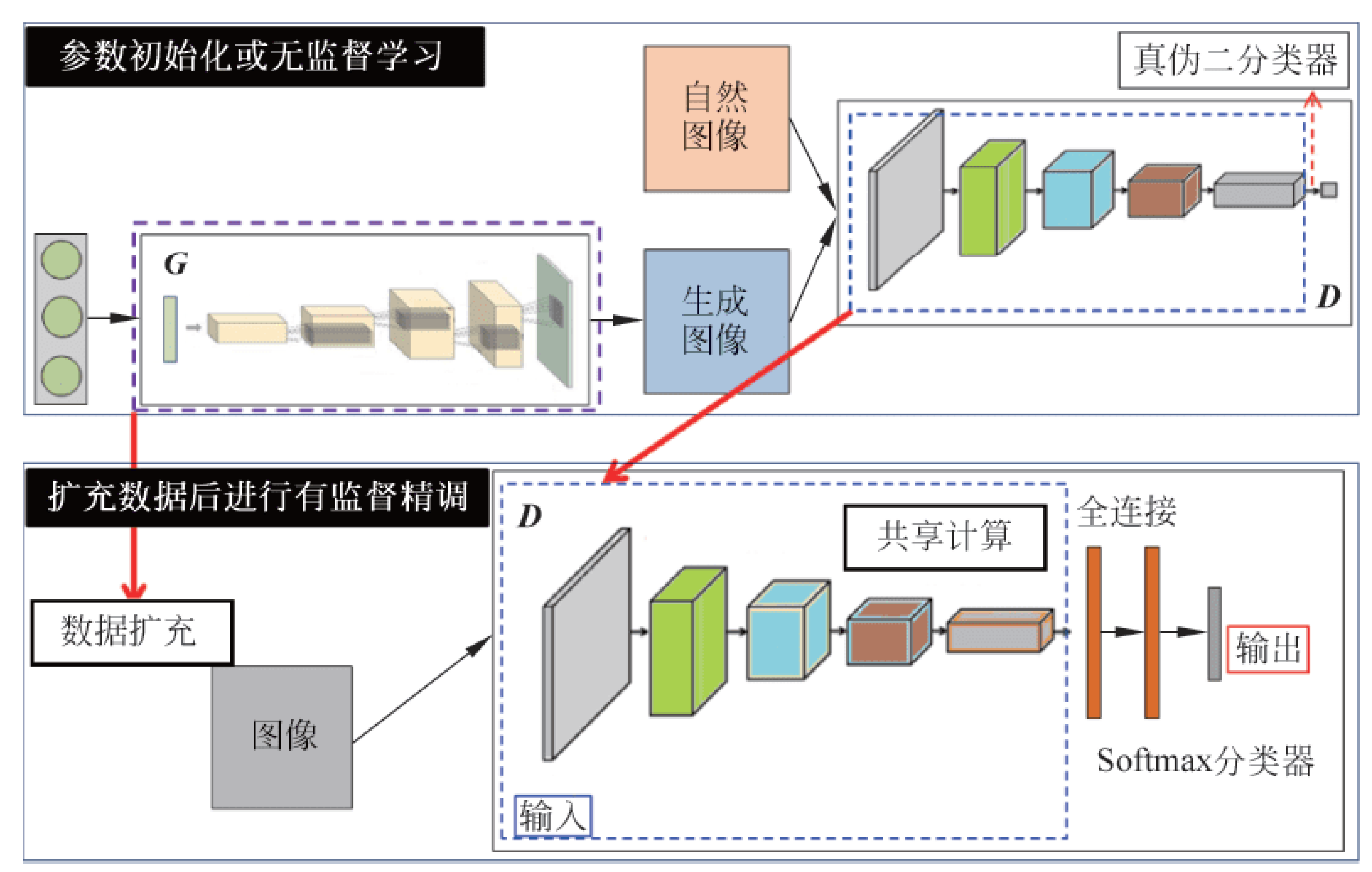


图 7.8　分类任务下生成式对抗网络的应用——数据扩充与共享计算

模型的参数量，以期保证生成数据的质量。

备注：关于生成式对抗网络的更多训练技巧将在第 11 章深度学习软件仿真平台及开发环境中的 Tensorflow 平台中给出。

7.2.3　网络模型的典型应用

对抗式生成网络已经在图像分类、检测、分割、高分辨率图像生成等诸多领域取得了突破性的成绩。但是它也存在一些问题，首先，它与传统的机器学习和深度学习方法一样，通常，假设训练数据与测试数据服从同样的分布，或者是在训练数据上的预测结果与在测试数据上的预测结果服从同样的分布。而实际上这两者存在一定的偏差，例如在测试数据上的预测准确率就通常比在训练数据上的要低，这就是过度拟合的问题。下面详细地陈述深度卷积生成对抗网络在分类任务、超分辨任务和分割任务上的应用以及对这一问题所做出的改进。

1. 分类任务

基于图 7.8，深度卷积生成对抗网络在无监督学习参数初始化阶段扮演着重要的角色。

1）数据（分类任务的数据集，包括训练集和测试集）

$$\begin{cases} \{\boldsymbol{x}_{\text{tr}}^{(n)}, \boldsymbol{t}^{(n)}\}_{n=1}^{N_{\text{tr}}} \longrightarrow \text{TrainData} \\ \{\boldsymbol{x}_{\text{te}}^{(n)}\}_{n=1}^{N_{\text{te}}} \longrightarrow \text{TestData} \end{cases} \tag{7.19}$$

这里 $\boldsymbol{t}^{(n)} \in \mathbb{R}^K$ 为期望输出，其类别个数为 K；除了分类任务的数据集外，深度卷积生成对抗网络的数据集为：



$$\{(\boldsymbol{x}^{(n)},\boldsymbol{z}^{(n)}),\boldsymbol{y}^{(n)}\}_{n=1}^{N} \tag{7.20}$$

注意这里的 $\boldsymbol{x}^{(n)}$ 为自然数据(或图像),它可以是分类任务中的训练数据或测试数据,也可以是二者的结合；不过这里使用的是训练数据；$\boldsymbol{z}^{(n)}$ 为服从某分布概型(例如归一化高斯分布、均匀分布等)下噪声的随机采样；另外 $\boldsymbol{y}^{(n)}$ 为输入$(\boldsymbol{x}^{(n)},\boldsymbol{z}^{(n)})$的真伪性判别向量,若为真(自然数据为输入),判别模型的输出为 1；若为伪,噪声数据通过生成模型获取的生成数据,再通过判别模型的输出为 0。

2）模型

与数据相对应,模型的设计也分为两个阶段：一是分类任务——深度卷积神经网络：

$$\begin{cases}\boldsymbol{X}_1 = \mathrm{ConvNet}(\boldsymbol{x},\theta^1)\\ \boldsymbol{X}_2 = \mathrm{FC}(\boldsymbol{X}_1,\theta^2)\\ \boldsymbol{t} = \mathrm{Softmax}(\boldsymbol{X}_2,K,\theta^3)\end{cases} \tag{7.21}$$

待优化的参数分为三个部分$(\theta^1,\theta^2,\theta^3)$,对于精调阶段,可以端到端方式进行训练。注意这里 $\boldsymbol{X}_1$ 是通过卷积网络获取得到的特征图或特征向量,该卷积网络的设计与下面深度卷积生成对抗网络中判别模型(除去真伪二分类器部分)是一致的。$\boldsymbol{X}_2$ 是添加的全连接层,通过增加“层数”来提升网络模型的泛化性能,若不添加,则 $\boldsymbol{X}_2=\boldsymbol{X}_1$；最后利用 Softmax 分类器实现输入数据(图像)所对应特征的分类任务。

二是参数初始化——深度卷积对抗神经网络：

$$\begin{cases}G: \tilde{\boldsymbol{x}} = g(\boldsymbol{z},\theta^G)\\ D: \begin{cases}\begin{cases}\boldsymbol{X}_1 = \mathrm{ConvNet}(\boldsymbol{x},\theta^1)\\ \tilde{\boldsymbol{X}}_1 = \mathrm{ConvNet}(\tilde{\boldsymbol{x}},\theta^1)\end{cases}\\ \boldsymbol{y} = \begin{pmatrix}P(L(\boldsymbol{x}) = \mathrm{real} \mid \boldsymbol{X}_1,\theta^C)\\ P(L(\tilde{\boldsymbol{x}}) = \mathrm{real} \mid \tilde{\boldsymbol{X}}_1,\theta^C)\end{pmatrix} \in \mathbb{R}^2\end{cases}\end{cases} \tag{7.22}$$

其中 $L(\boldsymbol{x})$为输入 $\boldsymbol{x}$ 所对应的真伪,这里的 G 为生成模型,D 为判别模型。

备注：这两个部分共享的是卷积网络(即式(7.21)和式(7.22)中关于 $\boldsymbol{X}_1$ 的定义)。

3）优化目标函数

根据模型的描述,优化目标函数的设计同样分为两个阶段,首先是参数初始化阶段,为方便下面公式描述,对于判别器记为：

$$D(\boldsymbol{x},\theta^D) = P(L(\boldsymbol{x}) = \mathrm{real} \mid \mathrm{ConvNet}(\boldsymbol{x},\theta^1),\theta^C) \tag{7.23}$$

其中参数 $\theta^D=(\theta^1,\theta^C)$,进一步优化目标函数：

$$\min_{\theta^D}\min_{\theta^G} J(\theta^D,\theta^G) = \left\{-\frac{1}{N}\left[\begin{array}{l}\sum_{n=1}^{N}\log(D(\boldsymbol{x}^{(n)},\theta^D)) + \\ \sum_{n=1}^{N}\log(1-D(G(\boldsymbol{z}^{(n)},\theta^G),\theta^D))\end{array}\right]\right\} \tag{7.24}$$

其次,精调阶段的优化目标函数为：



$$\min J(\theta^1,\theta^2,\theta^3)=\frac{1}{N_{tr}}\sum_{n=1}^{N_{tr}}\text{loss}(\hat{\boldsymbol{t}}^{(n)},\boldsymbol{t}^{(n)})+\lambda\cdot R(\theta^1,\theta^2,\theta^3) \tag{7.25}$$

其中$\hat{\boldsymbol{t}}^{(n)}$为输入经过深度卷积神经网络的预测输出，这里的损失函数为交叉熵形式，另外$R(\theta^1,\theta^2,\theta^3)$为(权值矩阵)参数的正则化约束，常用基于能量的$\|\cdot\|_2^2$或$\|\cdot\|_F^2$等。

4）求解

关于优化目标函数(7.24)，本章前边已陈述，假设求得的参数记为：

$$(\theta^{*G},\theta^{*D})=(\theta^{*G},(\theta^{*1},\theta^{*C})) \tag{7.26}$$

根据生成模型的要求，生成数据的分布概型P_G与训练数据集的分布概型P_{data}尽可能是一致的(注意，若深度卷积生成对抗网络的输入(指自然数据)不为分类任务中的训练数据，则难以实现训练数据集的扩充与“伪类标”传递)，即

$$\begin{cases} d(P_G,P_{\text{data}})\leqslant\xi \\ \text{data}=\{\boldsymbol{x}_{\text{tr}}^{(n)}\}_{n=1}^{N_{tr}} \end{cases} \tag{7.27}$$

其中$d(\cdot)$为衡量两个分布概型之间的差异性，如KL散度、沃瑟斯坦距离等，当ξ越小，说明差异性越小；接下来，可以利用如下的公式对分类任务中的训练数据集进行扩充：

$$\tilde{\boldsymbol{x}}=g(\boldsymbol{z},\theta^{*G})\sim P_G \tag{7.28}$$

如何做伪类标传递呢？可以将训练数据集按照K个类别分为K个子数据集，对每一个子数据集进行深度卷积生成对抗网络的优化学习，得到每一类子数据集的分布概型，如$P_G^{(k)}(k=1,2,\cdots,K)$，同样利用式(7.28)来扩充这一类的数据，最终实现整个训练数据集的扩充。假设扩充后的训练数据集记为：

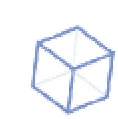
$$\{\boldsymbol{x}_{tr}^{(n)},\boldsymbol{t}_{tr}^{(n)}\}_{n=1}^{N_{tr}}\xrightarrow{\text{数据扩充}}\{\bar{\boldsymbol{x}}_{tr}^{(s)},\bar{\boldsymbol{t}}_{tr}^{(s)}\}_{s=1}^{S} \tag{7.29}$$

其中$S\geqslant N_{tr}$；接下来，关于优化目标函数(7.25)，利用该数据集，并通过共享计算的形式，将参数初始化得到的θ^{*1}赋给深度神经网络模型(7.21)中的θ^1(注意与之前的逐层学习策略不一样，这里是“特征”学习阶段的初始化)，另外目标函数中的个数N_{tr}改为S。求解过程与第4章深度卷积神经网络中描述的一致，这里不再赘述。

2. 超分辨任务

超分辨任务的核心是在寻找低分辨图像(高分辨图像的某种退化操作所得到的)与高分辨图像之间的关系，期望利用量化后的关系将低分辨图像通过恢复生动纹理和颗粒细节等以达到高分辨图像的过程，如图7.9所示。

与稀疏编码和传统深度神经网络方法(寻求在某一特征空间上，低分辨与高分辨图像所蕴含的特性是一致的)不同的是：深度卷积对抗生成网络(这里生成模型与判别模型的设计仍采用卷积架构，只不过生成模型是“上采样”的过程而判别模型是“下采样”的过程)使用零和博弈的策略在生成模型与判别模型之间寻求平衡(非合作纳什均衡点)。这里的生成网络采用具有16个残差模块的网络(残差网络仍为一种卷积架构)，而判别模型使用的是VGG网络(分类器为高低分辨率二分类器)，与稀疏编码和传统深度学习模型做图像超分辨率的



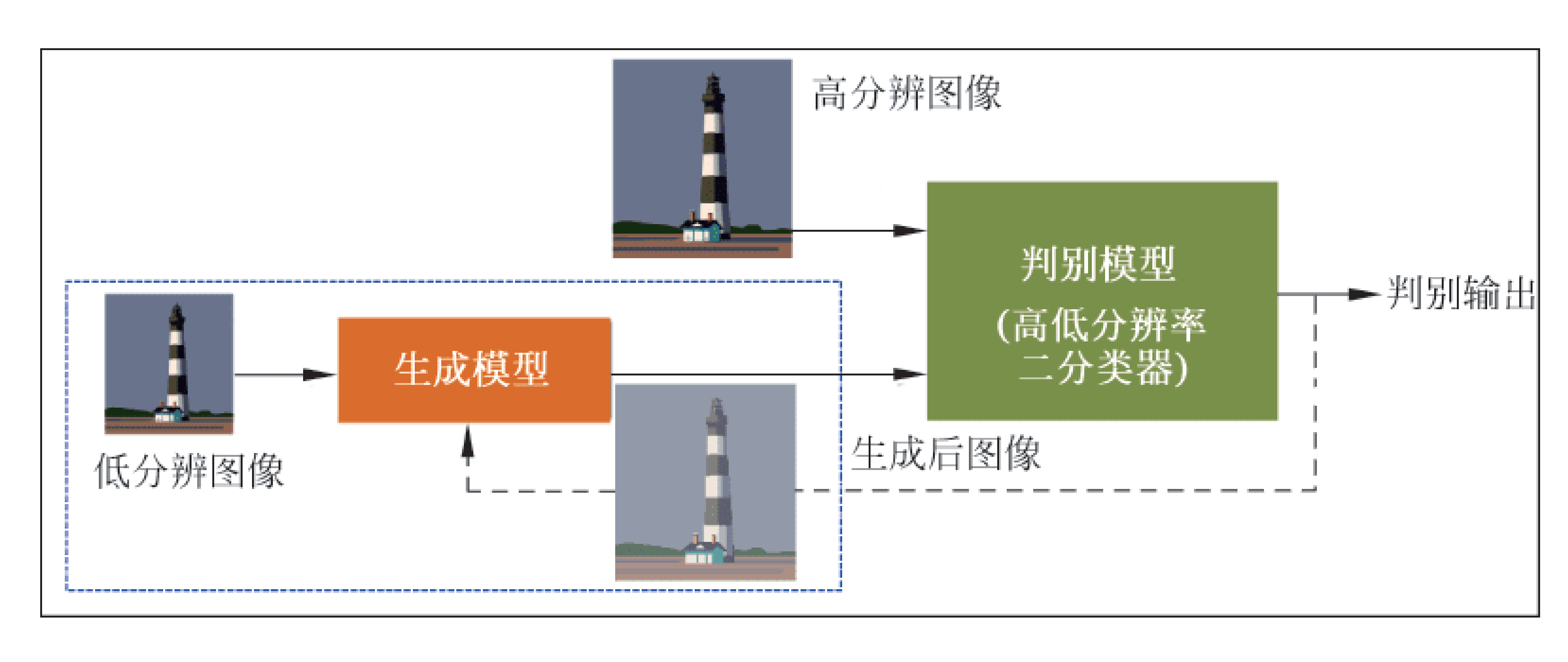


图 7.9　基于深度卷积生成对抗网络的超分辨任务实现

结果相比可以发现深度卷积生成对抗网络的结果能够提供更丰富的细节，这也是该模型做图像生成时（当整个网络模型收敛后，图 7.9 中虚线框部分）的一个显著优点，即能够提供更锐利的纹理和颗粒细节。

3. 分割任务

图像分割就是把图像分成若干个特定的、具有特定性质的区域并提出感兴趣目标的技术和过程。下面从数据、模型、优化目标函数和求解四个方面详述图 7.10 的过程。

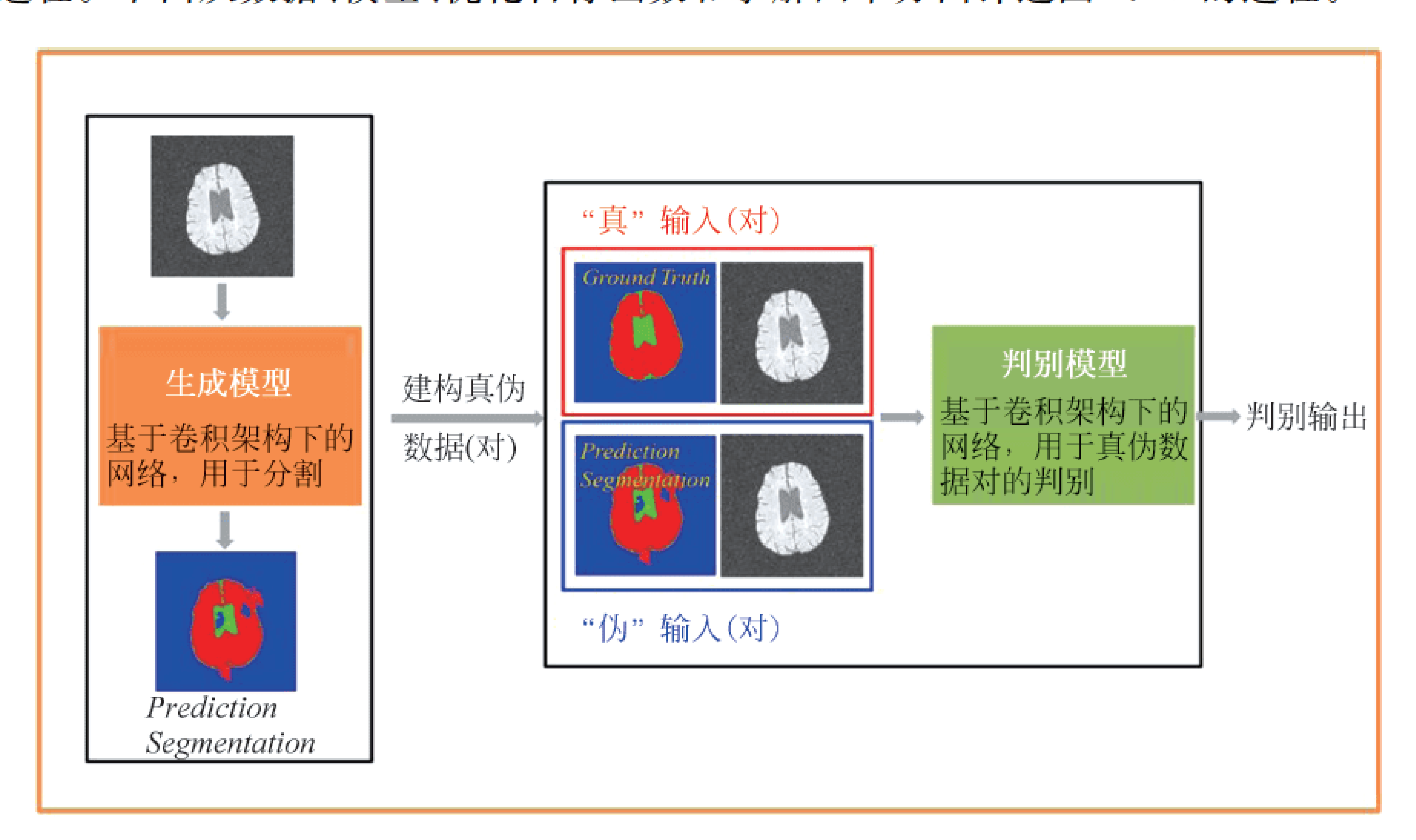


图 7.10　基于深度卷积生成对抗网络的分割任务实现

1）数据（分割任务训练数据集）

$$\{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N} \tag{7.30}$$

这里的 $\boldsymbol{x}^{(n)}$ 为输入图像，$\boldsymbol{y}^{(n)}$ 为期望（分割后）输出，即 Ground Truth。

2）模型

关于生成模型，主要描述式(7.30)中输入与输出之间的关系，可以采用传统的全卷积神经网络，即给定输入图像，得到预测或估计输出：

$$\hat{\boldsymbol{y}} = g(\boldsymbol{x}, \theta^G) \tag{7.31}$$

其中 $g(\cdot)$ 为生成模型，θ^G 为参数，$\hat{\boldsymbol{y}}$ 为 Ground Truth $\boldsymbol{y}$ 的逼近，记为预测分割结果 Prediction Segmentation。注意，这一部分的设计便是分割任务实现的核心。

关于判别模型，不同于之前的输入，这里该模型的输入为数据对，即利用式(7.31)可得：

$$\{((\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}), (\boldsymbol{x}^{(n)}, \hat{\boldsymbol{y}}^{(n)})), \boldsymbol{t}^{(n)} \in \mathbb{R}^2\}_{n=1}^{N} \tag{7.32}$$

其中真实输入为$(\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)})$，那么判别模型所对应的输出 $t^{(n)}(1)$ 为 1；生成输入为$(\boldsymbol{x}^{(n)}, \hat{\boldsymbol{y}}^{(n)})$，对应着 $\boldsymbol{t}^{(n)}(2)$ 为 0，其中 $\hat{\boldsymbol{y}}^{(n)}$ 为生成模型关于 $\boldsymbol{x}^{(n)}$ 的预测输出；如何设计该判别模型？

$$\boldsymbol{t} = \begin{pmatrix} d(\boldsymbol{x}, \boldsymbol{y}, \theta^D) \\ d(\boldsymbol{x}, \hat{\boldsymbol{y}}, \theta^D) \end{pmatrix} \tag{7.33}$$

其中 $d(\cdot)$ 为判别模型，θ^D 为参数且输出 $\boldsymbol{t}$ 真伪特性。

备注 1：判别模型的设计技巧，输入“对”数据可视为多通道并利用卷积神经网络处理，例如第 6 章深度多尺度几何网络，亦可设计“两条”通道，一条针对输入“对”中的 $\boldsymbol{x}$，另外一条针对输入“对”中的 y，之后在某层特征图后进行特征融合，进行全连接处理，最终通过真伪二分类器实现判别输出。

备注 2：判别模型的损失函数，通常由两部分构成，一部分是真实或自然数据集架构的，另一部分是生成或伪数据集架构的，与式(7.4)相同。

3）优化目标函数

依据数据集(7.32)，关于判别模型的优化目标函数为：

$$\min_{\theta^D}\left\{-\frac{1}{N}\left[\sum_{n=1}^{N}\log(d((\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}), \theta^D)) + \sum_{n=1}^{N}\log(1 - d((\boldsymbol{x}^{(n)}, \hat{\boldsymbol{y}}^{(n)}), \theta^D))\right]\right\} \tag{7.34}$$

其中的：

$$\hat{\boldsymbol{y}}^{(n)} = g(\boldsymbol{x}^{(n)}, \theta^G) \tag{7.35}$$

另外，根据生成模型(也称分割模型)的要求，一是预测分割结果尽可能与 Ground Truth 的差异性小，二是期望判别模型中真伪输入数据所对应的分布概型尽可能一致，所以得到生成模型的优化目标函数为：

$$\min_{\theta^G}\frac{1}{N}\sum_{n=1}^{N}\mathrm{loss}(\boldsymbol{y}^{(n)}, \hat{\boldsymbol{y}}^{(n)}) - \frac{1}{N}\sum_{n=1}^{N}\log(d((\boldsymbol{x}^{(n)}, g(x^{(n)}, \theta^G)), \theta^D)) \tag{7.36}$$

注意这里计算 θ^G 的时候，判别模型的参数 θ^D 是固定的，另外这里的 $\mathrm{loss}(\cdot)$ 可以使用基于能量的范数衡量，如 L_2 范数等。

4）求解

关于优化目标函数(7.34)和(7.36)的求解，仍采用交替迭代的优化方式，即固定判别模型的参数 θ^D，更新生成模型的参数 θ^G；同理来更新判别模型的参数 θ^D，最终使得整个网络达到稳定状态。

7.3 深度生成网络模型的新范式

7.3.1 生成式对抗网络的新范式

随着生成式对抗网络的提出,第一次从观念上挑战了传统的深度学习,改变了网络的设计及优化模式,通过引入零和博弈的理念,使得生成式对抗网络成为目前最为热门的深度学习框架,它的主要贡献在于:一是提供了一种全新的深度框架(该框架可容纳之前的深度神经网络);二是提供了一套严格的理论收敛性证明。另外对于它的改进,不仅有应用方面(如分类、分割和超分辨等),而且还有理论方面的突破,如基于最优化传输定理可以明确深度神经网络不再是"黑箱"操作等。下面主要从理论和应用方面简述生成式对抗网络的进展。

1. 理论方面

从数学角度,关于生成式对抗网络的收敛性描述为:假设生成模型和判别模型都有足够的性能的条件下,如果在迭代过程中的每一步,判别模型都可以达到当下在给定生成模型时的最优值,并在这之后再更新生成模型,那么最终生成数据的概率分布函数就一定会收敛于自然数据的概率分布函数。注意收敛性描述已经被严格的数学证明。不论是之前的损失敏感度生成式对抗网络 LS-GAN,还是沃瑟斯坦生成式对抗网络 W-GAN,其改进的核心在于克服判别模型具有无限可分能力时,生成模型容易出现梯度弥散的现象。下面将从距离角度和能量角度分别对生成式对抗网络进行改进,即主要陈述 F-GAN 和 EB-GAN 模型。

1) F-GAN(divergence family GAN)

原始的生成式对抗网络强调:假设已知自然数据集的分布概型为 P,通过生成模型估计出的分布概型为 Q,那么通过詹森香农散度(Jason Shannon Divergence)来刻画这两个分布概型之间的差异性,即

$$\begin{cases} D_{\mathrm{JS}}(P \parallel Q) = \dfrac{1}{2}D_{\mathrm{KL}}\left(P \parallel \dfrac{1}{2}(P+Q)\right) + \dfrac{1}{2}D_{\mathrm{KL}}\left(\dfrac{1}{2}(P+Q) \parallel Q\right) \\ D_{\mathrm{GAN}} = 2D_{\mathrm{JS}} - \log(4) \end{cases} \tag{7.37}$$

这里的 D 为散度,即 Divergence,而不是判别模型;D_{JS} 为詹森香农散度,D_{KL} 为 KL 散度和 D_{GAN} 为生成式对抗网络的散度。注意自然数据和生成数据所对应分布概型之间的差异性,主要体现在生成模型的要求,为了将詹森香农散度推广至一般的散度,提出了 F-GAN,其中的 F 可理解为所有散度的 family。

$$D_f(P \parallel Q) = \int_{\mathbb{R}} q(x) \cdot f\left(\frac{p(x)}{q(x)}\right)\mathrm{d}x \tag{7.38}$$

注意不同的 $f(\cdot)$对应着不同的散度(本质上,散度也是一种距离),常用的有函数 $f(u)=u\cdot\log(u)$(对应 KL 散度),$f(u)=(u-1)^2$(对应 Pearson 散度)以及 $f(u)=-(u+1)\log\left(\frac{1+u}{2}\right)+$

$u\log(u)$(对应詹森香农散度)。F-GAN 可以视为传统生成式对抗网络的一种推广,即刻画分布概型差异性的 D_{GAN} 所对应的生成式对抗网络是 F-GAN 的一种特例。

2) EB-GAN(Energy-Based GAN)

生成式对抗网络中判别模型的设计要求是:最大化区分或判别自然数据与生成数据,即将自然数据判断为真的概率或置信度要高,同时将生成数据判断为真的置信度要低;Lecun 研究团队发现,该要求等价于将判别模型视为一个能量函数,在自然数据集上能量函数所具有的能量值越低越好,在生成数据集上能量值则越高越好。依据新要求,他们改进了判别模型的设计(即网络结构),同时也给予生成式对抗网络一种能量模型的解释,即生成模型是以产生能量最小的样本为目的,而判别模型则以对这些产生的样本赋予较高的能量为目的。通常,从能量模型的角度来看待判别模型的好处是:可以用更多更宽泛的结构和损失函数来训练整个生成式对抗网络。下面我们给出一种基于自编码网络的判别模型来设计生成式对抗网络的结构,如图 7.11 所示。

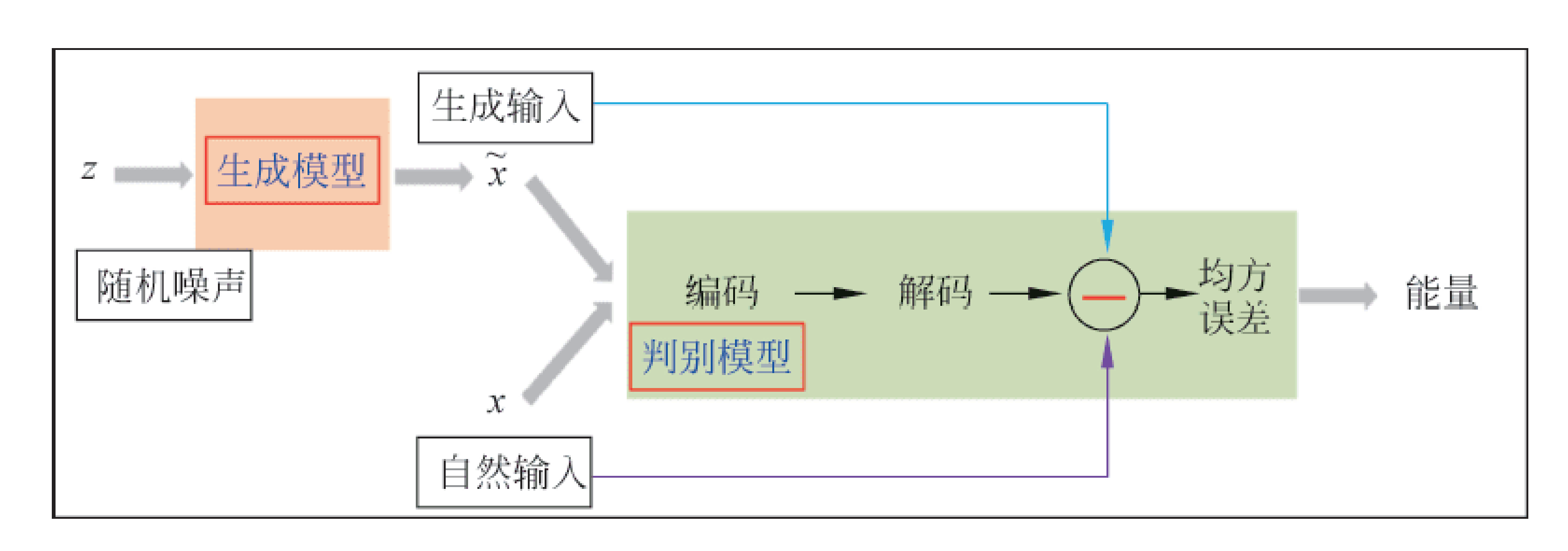


图 7.11　基于能量(判别模型)的生成式对抗网络模型

该模型的核心是理解判别模型针对自然输入(数据)的能量值越低,而对应着生成输入(数据)的能量值越高,如何从数学上刻画这一点?

假设自然输入(数据)集为

$$\{\boldsymbol{x}^{(n)}\}_{n=1}^{N} \tag{7.39}$$

这里的判别模型指通过自然数据集来训练,即

$$\min_{\theta} \frac{1}{N}\sum_{n=1}^{N} \| \hat{\boldsymbol{x}}^{(n)} - \boldsymbol{x}^{(n)} \|_2^2 + \lambda \cdot R(\theta) \tag{7.40}$$

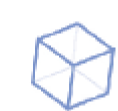
其中

$$\begin{cases} \hat{\boldsymbol{x}}^{(n)} = \text{Dec}(\text{Enc}(\boldsymbol{x}^{(n)}, \theta_1), \theta_2) \\ \theta = (\theta_1, \theta_2) \end{cases} \tag{7.41}$$

其中 Enc(•)表示编码函数,待优化的参数为 θ_1;Dec(•)为解码函数,待优化的参数为 θ_2。关于自编码理论已在第 4 章深度堆栈网络中详细介绍过,这里不再赘述。物理解释:由式(7.40)训练得到的判别模型在自然数据集上,其(损失)能量值:

$$\text{Energy}(\boldsymbol{x}, \theta) = \| \hat{\boldsymbol{x}} - \boldsymbol{x} \|_2^2 \tag{7.42}$$

满足“低”的特性(类似于“真”样本);注意判别模型的参数 θ 固定,对于生成数据,则相对



(损失)能量值满足“高”的特性(类似“伪”样本)。

备注：关于生成模型仍沿用传统的思路，例如(转置)卷积神经网络架构，本质上为随机噪声“上采样”逼近自然数据的过程。

2. 应用方面

已在本章7.2.3节介绍了生成式对抗网络的几种应用，包括分类、分割和图像的超分辨任务，这里介绍一个典型偏应用方面的工作，即感知相似度评价，目的是期望改进经典的生成式对抗网络关于生成图像清晰度的要求。众所周知，生成模型的要求是期望生成数据的分布概型尽可能接近自然数据的分布概型，所以对应的优化目标函数为：

$$J_G(\theta^G) = -\sum_{n=1}^{N} \log(D(G(z^{(n)}, \theta^G), \theta^D)) \tag{7.43}$$

其中 $z^{(n)}$ 为服从某分布(归一化高斯分布)下的随机采样，$n=1,2,\cdots,N$；另外符号 D 和 G 分别表示判别器和生成器，最小化优化式(7.43)时，判别模型的参数 θ^D 是固定的。判别模型的要求是遵循最大化判别准则，即

$$\begin{cases} J_D(\theta^D) = -\left[\sum_{n=1}^{N} \log(D(x^{(n)}, \theta^D)) + \sum_{n=1}^{N} \log(1 - D(\hat{x}^{(n)}, \theta^D))\right] \\ \hat{x}^{(n)} = G(z^{(n)}, \theta^G) \end{cases} \tag{7.44}$$

注意这里噪声数据的量与自然数据的量一致，实际应用中无须一致，优化判别模型时，生成模型的参数 θ^G 是固定的。

联合优化目标函数(7.43)和(7.44)便可达到非合作下的纳什均衡解，但为了提升生成图像的清晰度，引入感知相似度，即满足两方面要求：一是期望生成的图像尽可能与自然数据一致(注意随机噪声的量需满足与自然数据的量一样)；二是生成数据(图像)和自然数据在某特征域上能量尽可能一致；基于此得到如下的两个损失函数：

$$J_{\text{Image}}(\theta^G) = \sum_{n=1}^{N} \| G(z^{(n)}, \theta^G) - x^{(n)} \|_2^2 \tag{7.45}$$

$$J_{\text{Feature}}(\theta^G) = \sum_{n=1}^{N} \| T(G(z^{(n)}, \theta^G)) - T(x^{(n)}) \|_2^2 \tag{7.46}$$

注意关于生成式对抗网络的优化，仍为两个阶段，一是判别模型的优化仍采用式(7.44)，其中生成模型的参数固定；二是生成模型的优化为式(7.43)、式(7.45)和式(7.46)的加权和，通过拉格朗日乘子来平衡各损失项，即

$$\begin{aligned} &\min_{\theta^G} \lambda_1 \cdot J_G(\theta^G) + \lambda_2 \cdot J_{\text{Image}}(\theta^G) + \lambda_3 \cdot J_{\text{Feature}}(\theta^G) \\ &\text{s.t.} \begin{cases} \lambda_1 + \lambda_2 + \lambda_3 = 1 \\ 0 \leqslant \lambda_i \leqslant 1, i = 1,2,3 \end{cases} \end{aligned} \tag{7.47}$$

从目前的文献来看，生成式对抗网络主要应用在图像处理(图像超分辨、交互式图像生



成、图像翻译等)、自然语言处理(根据文本生成图像、对话生成和文本生成)等方面；另外，由于强化学习注重网络系统与环境交互作用下的“赏罚”刺激，所以结合强化学习的生成式对抗网络更易获得稳定且收敛的解，同时为深度强化学习注入新的模式。篇幅所限，应用介绍不一一展开。

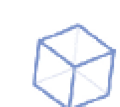
7.3.2 网络框架的性能分析与改进

1. 生成式对抗网络的本征分析

生成式对抗网络的稳定状态受限于自然(真实)数据的质量，以及交替优化算法的设计(设计使用何种距离来度量分布概型和真实概型之间的差异性)等的影响，同时生成模型与判别模型的参数初始化策略也可以加速整个网络的收敛进程。目前，关于“深度神经网络泛化性能的本源是什么”成为热(争)议的、且本质的焦点，例如谷歌大脑的 Samy Bengio 及其合作团队认为，网络的泛化性能源自于模型的容纳能力，即网络模型的有效容量对整个数据集的暴力“记忆”是足够大的，简言之，网络模型具有“记忆”特性。而加拿大蒙特利尔大学的 Yoshua Bengio 及其团队认为来源于正则化，不论是显式正则(如数据扩展、稀疏正则和权值衰减等)，还是隐式正则(迭代次数早终止策略、DropOut 和随机梯度下降求解方法等)，都可以提升网络的泛化性能。另外，二者都提出相应的论据互相弱化对方的观点。飞机能飞并不需要像鸟一样振翅，网络模型的泛化能力也并不全在网络“记忆”，就应用实践来看，更多的正则化策略及技巧(如参数初始化)能大幅度提升网络的性能，另外需要强调的是数据的数量并不是网络收敛或呈稳定状态的核心，还在于网络模型对每个数据(拓扑结构、分辨特性不同)的敏感度，即数据的“质”也非常的重要。生成式对抗网络不仅是一种“全新”的网络模型，更是一种全新的“框架”，在该框架下，从理论到应用实践都获取了诸多良好的物理解释和改进的思路及训练技巧，更是对“深度神经网络泛化性能的本源”做出了合理的回答，即网络呈现收敛或稳定的状态在理论上是通过零和博弈支撑的，在整个网络中(包含两个网络，即生成网络与判别网络)，一个网络的“收益”(即泛化性能的提升)是以另一个网络的“损失”为代价的，简言之，模型泛化性能的提升在于“负面”刺激或“批评”与监督(获得足够的经验能够自我反馈批评之前，有一个外部的批评家来纠正你每一小步的错误可以更容易训练生成网络及提升其泛化性能)。

2. 生成式对抗网络的历史沿承与发展

生成式对抗网络是深度生成网络的一种，在此之前就有将判别模型与生成模型进行联合学习的想法，例如 Tony Jebara 在 2001 年的毕业论文中就以最大熵的形式将判别模型与生成模型结合起来联合学习；2007 年 Zhuowen Tu 也提出将基于 boosting 分类器的判别模型与基于采样的生成模型相结合，来产生出服从真实分布的样本；2012 年 Jun Zhu 将最大间隔机制与贝叶斯模型相结合进行产生式模型的学习。与这些模型相比，2014 年 Ian



Goodfellow 等人提出的生成式对抗网络更加迎合了当下大数据的需求和深度学习的热潮，并且更重要的是它给出了一个大的框架及理论收敛性分析。在此框架的基础上，近三年来，从理论角度进行改进提出了 W-GAN、LS-GAN 等模型；从应用实践方面，提出的改进模型有 EB-GAN、DC-GAN、VAE-GAN 等；更多的关于生成式对抗网络的模型一一列举如下：

- GAN-Ian Goodfellow，2014-06
- DC-GAN-Alec Radford & Luke Metz，2015-11
- C-GAN-Mehdi Mirza，2014-11
- LAP-GAN-Emily Denton & Soumith Chintala，2015-06
- Info-GAN-Xi Chen，2016-06
- PP-GAN-Anh Nguyen，2016-12
- W-GAN-Martin Arjovsky，2017-01
- LS-GAN-Guo-Jun Qi，2017-01
- Seq-GAN-Lantao Yu，2016-09
- EB-GAN-Junbo Zhao，2016-09
- VAE-GAN-Anders Boesen Lindbo Larsen，2015-12
- Stacked GAN-Zhang H，2016
- Dual GAN-Zili Yi，2017

3. 网络框架的突破与改进

生成式对抗网络提供了一种无监督学习范式下的网络框架并给出了强有力的理论收敛性分析(非合作纳什均衡)，它为深度学习系统带来了极强的可塑性和扩展性，同时也包容了传统的机器学习理念(各种经典算法的移植)。当然该网络也存在着稳定性问题，以及客观量化评估(目前生成样本的质量仍依赖主观视觉去判断)；另外，从实践应用角度来说，目前多数方法或改进模型都是在原始框架的基础上稍做修改，例如修改损失函数，或者在 C-GAN 或 LAP-GAN 的基础上改进，仍没有一个具有突破、压倒性的图像生成模型，可能这和生成式对抗网络缺乏客观的评估指标有关；综合以上这些问题，该网络仍急需理论层面的进一步分析与支撑，同时在实践应用中总结更多可以保证稳定性的高效可行性策略及技巧。

7.4 应用驱动下的两种新生成式对抗网络

7.4.1 堆栈生成式对抗网络

问题描述：众所周知，通过文本描述来得到实际场景图像一直是计算机视觉中的难题，目前，诸多已知的处理方式所得到的图像仅能够粗略地反映文本所描述的意思，但常失效于

必要的细节和清晰的目标。

为了有效地解决这一实际应用问题，Han Zhang 等人提出了堆栈生成式对抗网络，模型主要包括两个阶段，第一阶段基于条件生成式对抗网络（C-GAN），依据文本的描述绘制目标的基本颜色和初始形状，从而产生低分辨图像的描述；第二阶段同样基于条件生成式对抗网络（注意与第一阶段的网络设计不同），将文本描述与第一阶段的输出作为该阶段的输入，以获取高分辨的图像，以期弥补细节或精细化处理。下面将详述这两个阶段并就网络模型的设计给出数学分析。

第一阶段、基于条件生成式对抗网络的低分辨图像描述如图 7.12 所示。

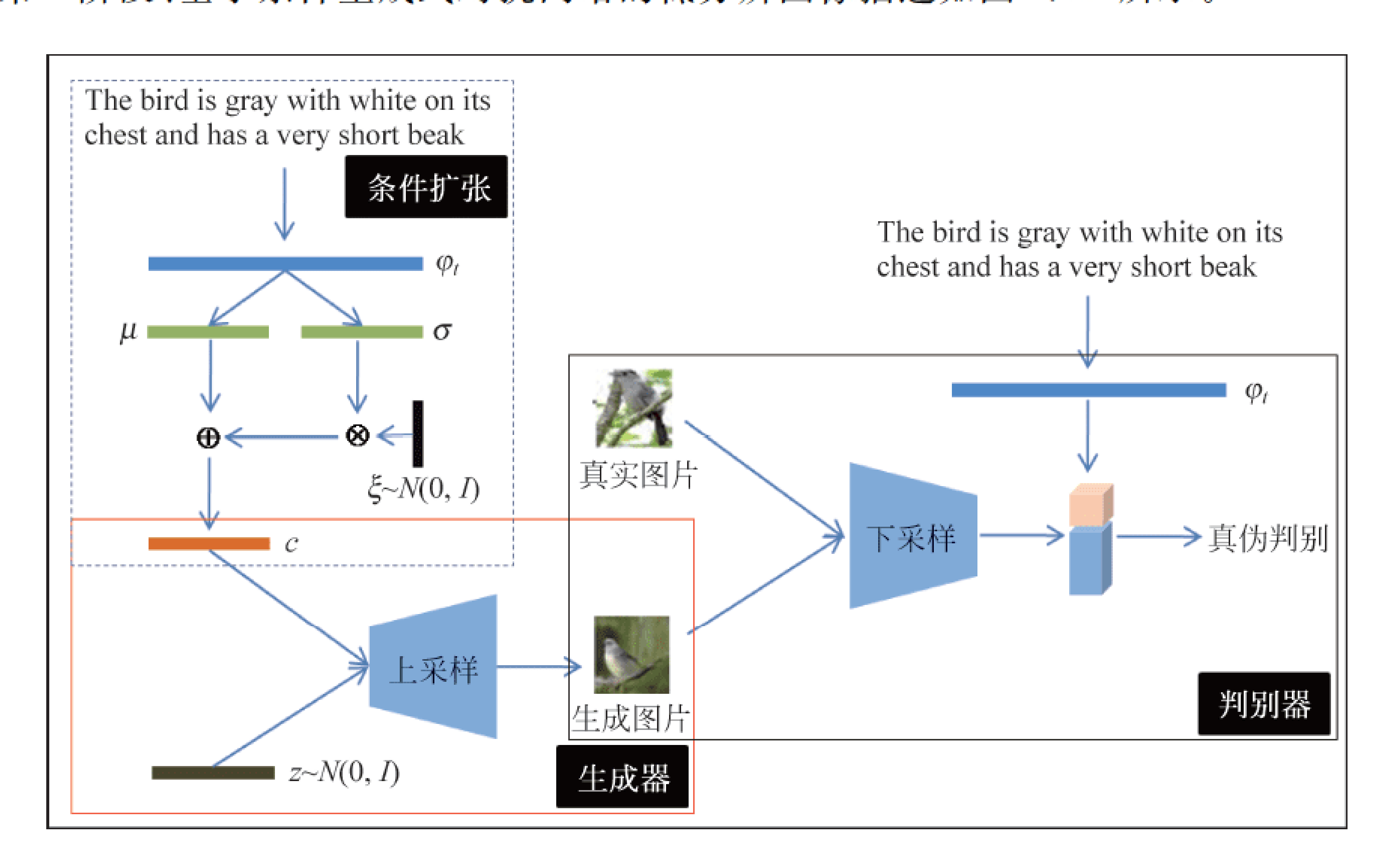


图 7.12 基于条件生成式对抗网络的低分辨图像描述

1. 数据

$$\{\boldsymbol{s}_n, \boldsymbol{z}_n, \boldsymbol{x}_n\}_{n=1}^{N} \tag{7.48}$$

其中 $\boldsymbol{s}_n$ 为第 n 幅低分辨图像 $\boldsymbol{x}_n$ 的文本描述，另外 $\boldsymbol{z}_n$ 为服从标准高斯分布下的随机噪声。

2. 模型

这一阶段模型的架构分为三块，第一块为条件扩张：

$$\begin{cases} \boldsymbol{\varphi}_t = P(\boldsymbol{s}_n) \\ (\boldsymbol{\mu}, \boldsymbol{\sigma}) = K(\boldsymbol{\varphi}_t) \\ \boldsymbol{c} = \boldsymbol{T}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \boldsymbol{\xi}) \end{cases} \tag{7.49}$$

这里的符号“P, K, T”均为相应的统计操作，其中 $\boldsymbol{\varphi}_t$ 为将文本转化为词向量，$\boldsymbol{c}$ 为条件表征。

第二块与第三块分别为生成器、判别器的设计，其仍可以采用深度卷积生成式对抗网络的架构，这里不再赘述。注意在判别器的设计中，需要加入词向量$\boldsymbol{\varphi}_t$，经过压缩与空域复制操作进行向量矩阵化处理，从而实现文本信息的有效嵌入。

3. 优化目标函数

$$\begin{aligned}&\min_G \max_D V(D,G,\boldsymbol{c},\boldsymbol{\varphi}_t)\\&=\mathrm{E}_{x\sim P_{\text{data}}}[\log(D(\boldsymbol{x},\boldsymbol{\varphi}_t,\theta_D))]+\mathrm{E}_{z\sim P_z}[\log(1-D(G(\boldsymbol{z}_n,\boldsymbol{c},\theta_G),\boldsymbol{\varphi}_t,\theta_D))]\end{aligned}\tag{7.50}$$

注意这里的$\boldsymbol{\varphi}_t$ 和 $\boldsymbol{c}$ 分别为文本所对应的词向量和条件表征。另外，在实际应用中为了获取较为平滑的图像，同时为了避免过拟合现象，通常在训练阶段，对生成器加入如下的正则项：

$$D_{KL}(N(\boldsymbol{\mu}_{\varphi_t},\boldsymbol{\sigma}_{\varphi_t})\parallel N(\boldsymbol{0},\boldsymbol{I}))\tag{7.51}$$

其中 D_{KL} 为 KL 散度；故关于判别器与生成器的优化目标函数为：

$$\begin{cases}L_D=-\dfrac{1}{N}\left[\displaystyle\sum_{n=1}^{N}\log(D(\boldsymbol{x}_n,\boldsymbol{\varphi}_{t,n},\theta_D))+\sum_{n=1}^{N}\log(1-D(G(\boldsymbol{z}_n,\boldsymbol{c}_n,\theta_G),\boldsymbol{\varphi}_{t,n},\theta_D))\right]\\L_G=\dfrac{1}{N}\left[\displaystyle\sum_{n=1}^{N}\log(1-D(G(\boldsymbol{z}_n,\boldsymbol{c}_n,\theta_G),\boldsymbol{\varphi}_{t,n},\theta_D))\right]+\lambda\cdot D_{KL}(N(\boldsymbol{\mu}_{\varphi_t},\boldsymbol{\sigma}_{\varphi_t})\parallel N(\boldsymbol{0},\boldsymbol{I}))\end{cases}\tag{7.52}$$

注意$\boldsymbol{\varphi}_{t,n}$为第 n 条文本描述所对应的词向量，$\boldsymbol{x}_n$ 为第 n 条文本所对应的低分辨图像，$\boldsymbol{c}_n$ 为对应的条件表征。

4. 求解

关于优化目标函数(7.52)，通过交替迭代更新算法，实现生成器与判别器中参数的求解。

第二阶段为基于条件生成式对抗网络的高分辨图像精修。

根据图 7.13 知，此时生成器的输入为文本描述所对应的条件表征 $\boldsymbol{c}$ 和低分辨生成图像(即第一阶段生成器的输出)；本阶段的核心描述类似于 7.2.3 节中超分辨任务的描述，与之不同的是，网络设计中融入应用任务所刻画的词向量与条件表征。另外，需要注意的是这一阶段的数据包括

$$\{\boldsymbol{s}_n,\boldsymbol{x}_n,\tilde{\boldsymbol{x}}_n\}_{n=1}^{N}\tag{7.53}$$

其中$\tilde{\boldsymbol{x}}_n$ 为低分辨图像 $\boldsymbol{x}_n$ 所对应的高分辨图像。仿照第一阶段的公式(7.52)，关于判别器的优化目标函数为：

$$L_{\widetilde{D}}=-\frac{1}{N}\left[\sum_{n=1}^{N}\log(\widetilde{D}(\tilde{\boldsymbol{x}}_n,\boldsymbol{\varphi}_{t,n},\theta_{\widetilde{D}}))+\sum_{n=1}^{N}\log(1-\widetilde{D}(\widetilde{G}(\boldsymbol{x}_n,\boldsymbol{c}_n,\theta_{\widetilde{G}}),\boldsymbol{\varphi}_{t,n},\theta_{\widetilde{D}}))\right]\tag{7.54}$$

关于生成器的优化目标函数为：

$$L_{\widetilde{G}}=\frac{1}{N}\left[\sum_{n=1}^{N}\log(1-\widetilde{D}(\widetilde{G}(x_n,\boldsymbol{c}_n,\theta_{\widetilde{G}}),\boldsymbol{\varphi}_{t,n},\theta_{\widetilde{D}}))\right]+\lambda\cdot D_{KL}(N(\boldsymbol{\mu}_{\varphi_t},\boldsymbol{\sigma}_{\varphi_t})\parallel N(\boldsymbol{0},\boldsymbol{I}))\tag{7.55}$$

关于优化求解可以参考 7.3.1 节中生成式对抗网络的新范式，这里不再赘述。

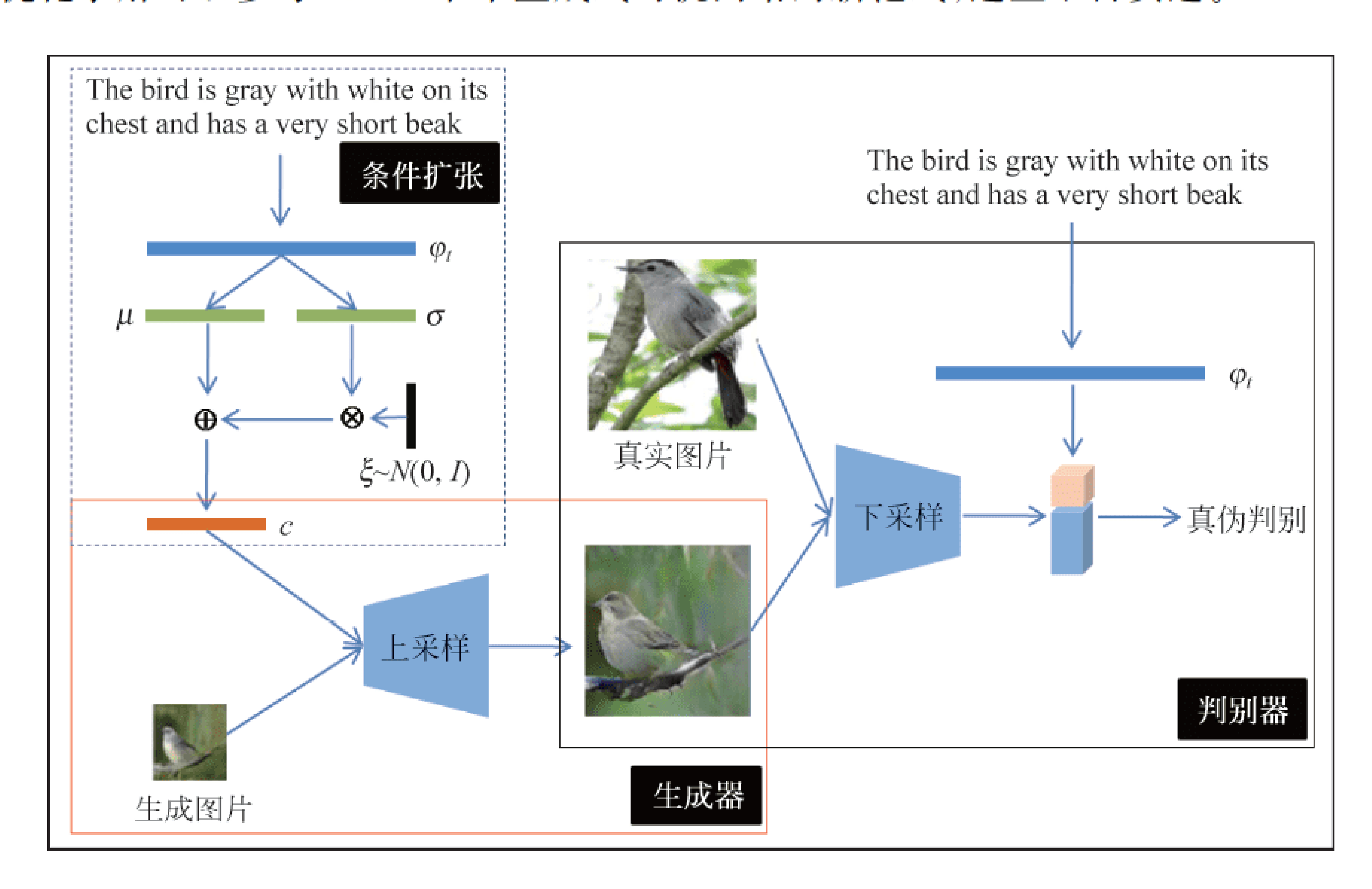


图 7.13　基于条件生成式对抗网络的高分辨图像精修

7.4.2　对偶学习范式下的生成式对抗网络

从 2016 年开始，无监督学习进入实质性发展阶段，逐层学习、生成式对抗网络等为无监督方式下的深度学习注入新的活力，同时对偶学习也为研究过程中所遇到的困难提供了新的思路。针对机器“翻译”任务，包括图片不同风格间的转化、不同语种间的翻译等，对偶学习范式下的生成式对抗网络，充分利用未标注的数据，提高对偶任务中的两个“翻译”模型的性能，如图 7.14 所示。下面针对图片不同风格间的转化任务(如彩色照片与素描图之间的转化)给予详细的描述和数学分析。

1. 数据(训练数据)

$$\{\boldsymbol{s}_n,\boldsymbol{x}_n\}_{n=1}^{N}\tag{7.56}$$

其中 $\boldsymbol{s}_n$ 为素描图，$\boldsymbol{x}_n$ 为素描图所对应的图片。

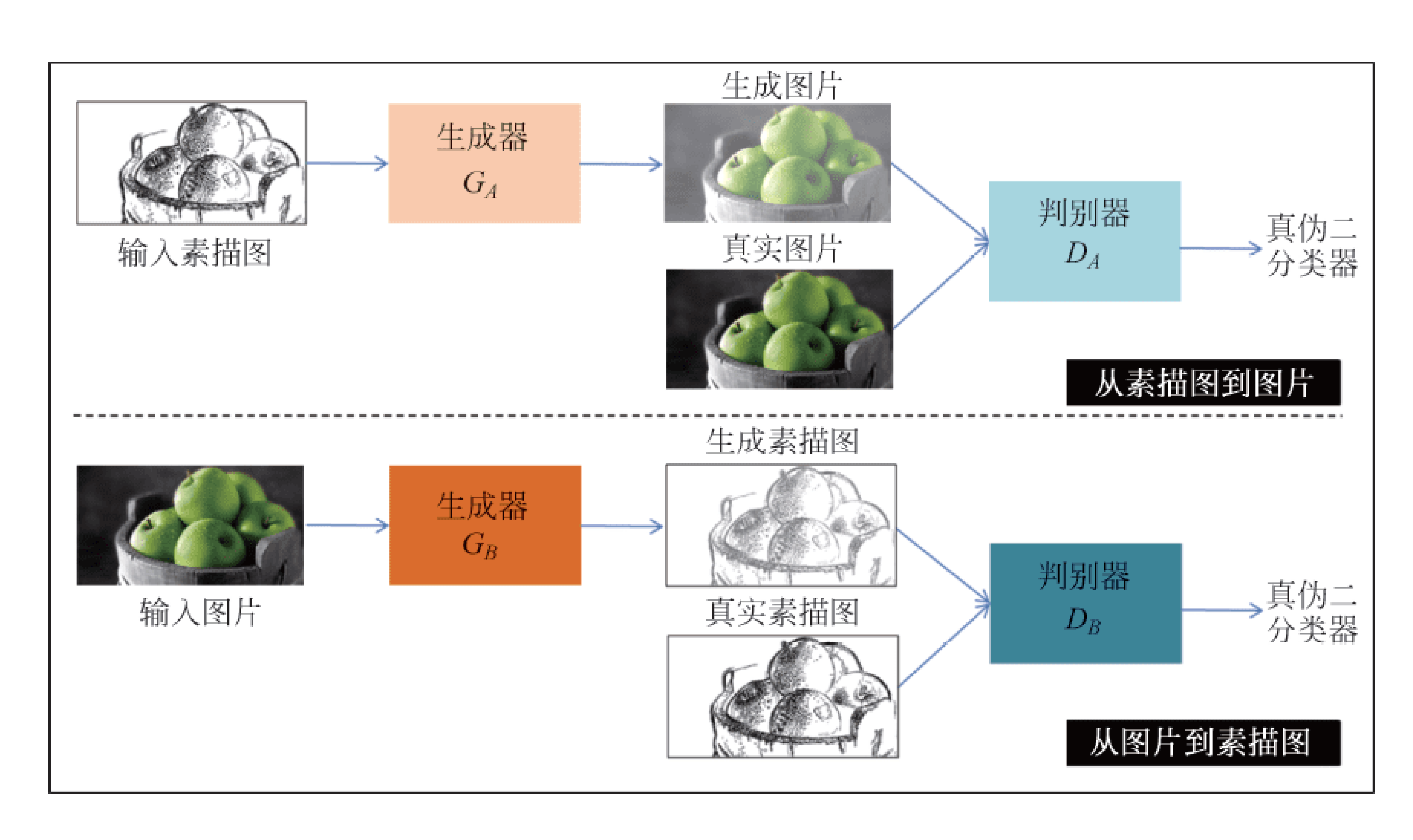


图 7.14　基于对偶生成式对抗网络的图片风格转化任务

2. 模型与优化目标函数

模型的设计包括两个模块，分别是从素描图到图片模块和从图片到素描图模块，前者将素描图作为输入，真实图片作为监督并修正生成的图片，即优化目标函数为：

$$\min_{\theta_{G_A}} \max_{\theta_{D_A}} V(G_A, D_A, s, x) = \mathrm{E}_x[\log(G_A(x, \theta_{G_A}))] + \mathrm{E}_s[\log(G_A(D_A(s, \theta_{D_A}), \theta_{G_A}))] \tag{7.57}$$

而后者将图片作为输入，真实素描图作为监督并修正生成的素描图，即优化目标函数为：

$$\min_{\theta_{G_B}} \max_{\theta_{D_B}} V(G_B, D_B, s, x) = \mathrm{E}_s[\log(G_B(s, \theta_{G_B}))] + \mathrm{E}_x[\log(G_B(D_B(x, \theta_{D_B}), \theta_{G_B}))] \tag{7.58}$$

注意，为了建立这两个模块之间的联系，要求两点，一是素描图的相似性，即满足：

$$\| G_B(G_A(\boldsymbol{s}, \theta_{G_A}), \theta_{G_B}) - \boldsymbol{s} \|_F^2 \leqslant \varepsilon \tag{7.59}$$

即素描图通过这两个模块的操作，使得误差最小；二是图片的相似性，即满足：

$$\| G_A(G_B(\boldsymbol{x}, \theta_{G_B}), \theta_{G_A}) - \boldsymbol{x} \|_F^2 \leqslant \varepsilon \tag{7.60}$$

即图片通过这两个模块的操作，使得误差最小。注意这一系统的稳态发生的前提是满足式(7.59)和式(7.60)，以及两个模块对应的判别器 D_A 和 D_B 满足最大化识别准则。

3. 求解

关于优化目标函数(7.57)和(7.58)，可采用交替迭代更新的方法，包括每一个模块

的参数更新，以及模块间带有约束(7.59)和(7.60)的迭代更新。详细请参考给出的论文。

7.5 变分自编码器

目前，深度生成模型的组成包括三部分，一是深度置信网络、深度玻尔兹曼机；二是生成式对抗网络及其变形；三是变分自编码器。注意与前两种模式不同的是，变分自编码器更注重隐特征空间中概率密度函数的学习与刻画。下面根据网络模型的结构(见图7.15)，进行详细的数学分析。

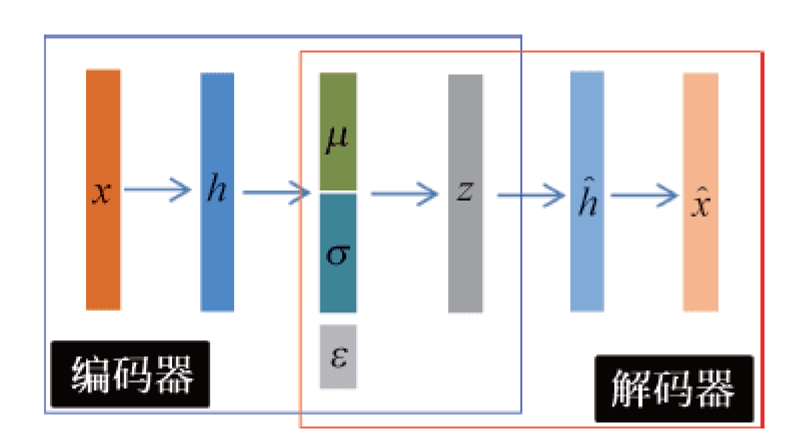


图7.15 变分自编码器的结构

注意图7.15中的$\boldsymbol{\varepsilon}$为噪声，为了下面描述的方便，简记编码器的概率生成方式为$P_{\text{Encoder}}(\boldsymbol{z}|\boldsymbol{x},\theta)$；解码器为$P_{\text{Decoder}}(\boldsymbol{x}|\boldsymbol{z},\vartheta)$；其中$\theta$和$\vartheta$为待学习的参数，则优化目标函数为：

$$\max_{\theta,\vartheta} \mathrm{E}_{P_{\text{Encoder}}(\boldsymbol{z}|\boldsymbol{x},\theta)}[\log(P_{\text{Decoder}}(\boldsymbol{x} \mid \boldsymbol{z},\vartheta))]$$
$$\text{s.t. } D_{\text{KL}}(P_{\text{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x},\theta) \parallel P(z)) \leqslant \varepsilon \tag{7.61}$$

等价地，拉格朗日增广函数为：

$$\max_{\theta,\vartheta} E_{P_{\text{Encoder}}(z|x,\theta)}[\log(P_{\text{Decoder}}(\boldsymbol{x} \mid \boldsymbol{z},\vartheta))] - \lambda \cdot D_{\text{KL}}(P_{\text{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x},\theta) \parallel P(\boldsymbol{z})) \leqslant \varepsilon \tag{7.62}$$

其中的$\lambda>0$为拉格朗日因子，另外这里的约束先验$P(z)$为标准的高斯分布，即

$$P(\boldsymbol{z}) \sim N(\boldsymbol{0},\boldsymbol{I}) \tag{7.63}$$

注意：通常在实际应用中，例如手写体数据，期望通过编码器，挖掘输入数据在隐特征空间上概率密度函数的有效表达，然后根据概率密度函数进行采样，通过解码器生成同类但风格迥异的手写体数据。首先，假设所有的手写体数据$\{\boldsymbol{x}_n\}_{n=1}^{N}$都是独立同分布的，则利用最大负对数似然法，对生成模型$P_{\text{Decoder}}(\boldsymbol{x}|\boldsymbol{z},\vartheta)$进行参数估计，即有如下的优化目标：

$$\begin{aligned}\max_{\vartheta} -\log(P_{\text{Decoder}}(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_N,\vartheta)) &= -\log\left(\prod_{n=1}^{N} P_{\text{Decoder}}(\boldsymbol{x}_n,\vartheta)\right) \\ &= -\sum_{n=1}^{N}\log P_{\text{Decoder}}(\boldsymbol{x}_n,\vartheta)\end{aligned} \tag{7.64}$$

注意利用变分自编码器中编码概型去逼近解码的后验概型，即

$$P_{\text{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta) \xrightarrow{\text{逼近}} P_{\text{Decoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\vartheta) \tag{7.65}$$

其中$n=1,2,\cdots,N$。进一步通过KL散度衡量二者之间的差异性，从而有

$$
\begin{aligned}
&D_{\mathrm{KL}}(P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta) \parallel P_{\mathrm{Decoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\vartheta)) \\
&= \mathrm{E}_{P_{\mathrm{Encoder}}(\boldsymbol{z}|\boldsymbol{x}_n,\theta)}\left[\log\left(\frac{P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta)}{P_{\mathrm{Decoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\vartheta)}\right)\right] \\
&= \mathrm{E}_{P_{\mathrm{Encoder}}(\boldsymbol{z}|\boldsymbol{x}_n,\theta)}\left[\log\left(\frac{P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta)}{P_{\mathrm{Decoder}}(\boldsymbol{z},\boldsymbol{x}_n,\vartheta)}\right)+\log(P_{\mathrm{Decoder}}(\boldsymbol{x}_n,\vartheta))\right]
\end{aligned} \tag{7.66}
$$

进一步，对于式(7.66)有：

$$
\begin{aligned}
\log(P_{\mathrm{Decoder}}(\boldsymbol{x}_n,\vartheta)) =& -\mathrm{E}_{P_{\mathrm{Encoder}}(\boldsymbol{z}|\boldsymbol{x}_n,\theta)}\left[\log\left(\frac{P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta)}{P_{\mathrm{Decoder}}(\boldsymbol{z},\boldsymbol{x}_n,\vartheta)}\right)\right] \\
&+ D_{\mathrm{KL}}(P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta) \parallel P_{\mathrm{Decoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\vartheta))
\end{aligned} \tag{7.67}
$$

注意式(7.67)与式(7.62)具有一致性，特别当 $\lambda=1$ 时，二者相等。

通常直接优化目标函数(7.64)是不可行的，因 KL 散度通常是小于零的，所以根据式(7.66)有：

$$
\begin{cases}
\log(P_{\mathrm{Decoder}}(\boldsymbol{x}_n,\vartheta)) \geqslant L(\theta,\vartheta,\boldsymbol{x}_n) \\
L(\theta,\vartheta,\boldsymbol{x}_n) \triangleq -\mathrm{E}_{P_{\mathrm{Encoder}}(\boldsymbol{z}|\boldsymbol{x}_n,\theta)}\left[\log\left(\frac{P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta)}{P_{\mathrm{Decoder}}(\boldsymbol{z},\boldsymbol{x}_n,\vartheta)}\right)\right] \\
\qquad = \mathrm{E}_{P_{\mathrm{Encoder}}(\boldsymbol{z}|\boldsymbol{x}_n,\theta)}[\log(P_{\mathrm{Decoder}}(\boldsymbol{z},\boldsymbol{x}_n,\vartheta)) - \log(P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta))]
\end{cases} \tag{7.68}
$$

进而将优化问题转化为它的下界：

$$
\min_{\theta,\vartheta} L(\theta,\vartheta) = \sum_{n=1}^{N} L(\theta,\vartheta,\boldsymbol{x}_n) \tag{7.69}
$$

由于 $L(\theta,\vartheta,\boldsymbol{x}_n)$关于参数 θ 的梯度方差很大，不适用于数值计算；为此有两种改进的思路：一种是将编码器的概率生成方式 $P_{\mathrm{Encoder}}(\boldsymbol{z}|\boldsymbol{x},\theta)$替代为 $\boldsymbol{z}=P(\varepsilon,\boldsymbol{x})$，其中的 ε 为噪声，即图 7.15 中的描述。于是得到优化目标函数(7.69)的估计式为：

$$
\widetilde{L}(\theta,\vartheta,\boldsymbol{x}_n) = \frac{1}{L}\sum_{l=1}^{L}[\log(P_{\mathrm{Decoder}}(\boldsymbol{x}_n,\boldsymbol{z}_{n,l},\vartheta)) - \log(P_{\mathrm{Encoder}}(\boldsymbol{z}_{n,l} \mid \boldsymbol{x}_n,\theta))] \tag{7.70}
$$

其中 L 为噪声分布 $P(\varepsilon)$下采样的个数，且

$$
\begin{cases}
\boldsymbol{z}_{n,l} = P(\varepsilon_l,\boldsymbol{x}_n) \\
\varepsilon_l \sim P(\varepsilon)
\end{cases} \tag{7.71}
$$

另一种思路是将 $P_{\mathrm{Decoder}}(\boldsymbol{x}_n,\vartheta)$替代为 $P_{\mathrm{Decoder}}(\boldsymbol{x}_n|\boldsymbol{z},\vartheta)$，以及将 $P_{\mathrm{Decoder}}(\boldsymbol{z}|\boldsymbol{x}_n,\vartheta)$替换为 $P(\boldsymbol{z})$，将其代入到式(7.66)中得到另一个优化目标函数的估计式为：

$$
\bar{L}(\theta,\vartheta,\boldsymbol{x}_n) = \frac{1}{L}\sum_{l=1}^{L}\log(P_\theta(\boldsymbol{x}_n \mid \boldsymbol{z}_{n,l},\vartheta)) - D_{\mathrm{KL}}(P_{\mathrm{Encoder}}(\boldsymbol{z} \mid \boldsymbol{x}_n,\theta) \parallel P(\boldsymbol{z})) \tag{7.72}
$$

其中 L 的解释与上面的一致，并且满足式(7.71)。注意在实际应用中，为了启动该算法，常取：

$$
\begin{cases}
P(\varepsilon) \sim N(\boldsymbol{0},\boldsymbol{I}) \\
P(\boldsymbol{z}) \sim N(\boldsymbol{0},\boldsymbol{I}) \\
\boldsymbol{z} = P(\varepsilon,\boldsymbol{x}) = \boldsymbol{\mu}+\boldsymbol{\sigma}\odot\varepsilon
\end{cases} \tag{7.73}
$$



具体地，根据图 7.15，模型的描述分为编码阶段与解码阶段，其中编码阶段的关系为：

$$\begin{cases} \boldsymbol{h} = \tanh(\boldsymbol{W}_1 \cdot \boldsymbol{x} + \boldsymbol{b}_1) \\ \boldsymbol{\mu} = \boldsymbol{W}_2 \cdot \boldsymbol{h} + \boldsymbol{b}_2 \\ \boldsymbol{\sigma} = \boldsymbol{W}_3 \cdot \boldsymbol{h} + \boldsymbol{b}_3 \\ \boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \varepsilon \end{cases} \tag{7.74}$$

解码阶段的关系为：

$$\begin{cases} \hat{\boldsymbol{h}} = \tanh(\boldsymbol{W}_4 \cdot \boldsymbol{z} + \boldsymbol{b}_4) \\ \hat{\boldsymbol{x}} = \tanh(\boldsymbol{W}_5 \cdot \hat{\boldsymbol{h}} + \boldsymbol{b}_5) \end{cases} \tag{7.75}$$

优化目标函数可以采用式(7.70)或式(7.72)进行求解。

参考文献

[7.1] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. Nature, 2015, 521(7553): 436-444.

[7.2] Han S, Mao H, Dally W J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding[J]. Fiber, 2016, 56(4): 3-7.

[7.3] Peng X B, Berseth G, Michiel V D P. Terrain-adaptive locomotion skills using deep reinforcement learning[J]. Acm Transactions on Graphics, 2016, 35(4): 81.

[7.4] Peng X B, Panne M V D. Learning Locomotion Skills Using DeepRL: Does the Choice of Action Space Matter? [J]. 2016.

[7.5] Szegedy C. An Overview of Deep Learning[J]. AITP 2016, 2016.

[7.6] Wu J, Zhang C, Xue T, et al. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling[J]. 2016.

[7.7] Mirza M, Osindero S. Conditional Generative Adversarial Nets[J]. Computer Science, 2014: 2672-2680.

[7.8] Goodfellow I J, Pougetabadie J, Mirza M, et al. Generative Adversarial Networks[J]. Advances in Neural Information Processing Systems, 2014, 3: 2672-2680.

[7.9] Gauthier J. Conditional generative adversarial nets for convolutional face generation[J]. Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester, 2014.

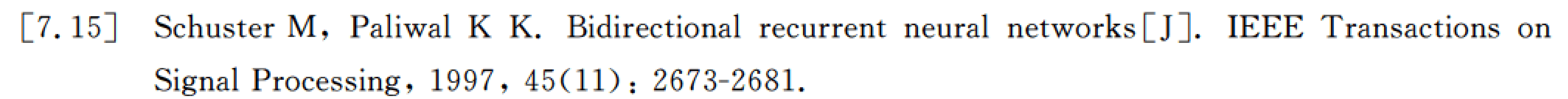
[7.10] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks[J]. Computer Science, 2015.

[7.11] Denton E, Chintala S, Szlam A, et al. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks[J]. Computer Science, 2015.

[7.12] Liu M Y, Tuzel O. Coupled Generative Adversarial Networks[J]. 2016.

[7.13] Chen M, Denoyer L. Multi-view Generative Adversarial Networks[J]. 2016.

[7.14] Arjovsky M, Bottou L. Towards Principled Methods for Training Generative Adversarial Networks[J]. 2017.



[7.15] Schuster M, Paliwal K K. Bidirectional recurrent neural networks[J]. IEEE Transactions on Signal Processing, 1997, 45(11): 2673-2681.

[7.16] Graves A. Supervised Sequence Labelling with Recurrent Neural Networks[M]. Springer Berlin Heidelberg, 2012.

[7.17] Graves A, Mohamed A R, Hinton G. Speech recognition with deep recurrent neural networks[J]. 2013, 38(2003): 6645-6649.

[7.18] Krueger D, Memisevic R. Regularizing RNNs by Stabilizing Activations [J]. Computer Science, 2015.

[7.19] Ji S, Vishwanathan S V N, Satish N, et al. BlackOut: Speeding up Recurrent Neural Network Language Models With Very Large Vocabularies [J]. Computer Science, 2015, 115 (8): 2159-2168.

[7.20] Choi E, Bahadori M T, Schuetz A, et al. Doctor AI: Predicting Clinical Events via Recurrent Neural Networks[J]. Computer Science, 2015.

[7.21] Talathi S S, Vartak A. Improving Performance of Recurrent Neural Network with ReLU Nonlinearity[J]. Computer Science, 2015.

[7.22] Karpathy A, Johnson J, Li F F. Visualizing and Understanding Recurrent Networks[J]. 2015.

[7.23] Hidasi B, Karatzoglou A, Baltrunas L, et al. Session-based Recommendations with Recurrent Neural Networks[J]. Computer Science, 2015.

[7.24] Bashivan P, Rish I, Yeasin M, et al. Learning Representations from EEG with Deep Recurrent-Convolutional Neural Networks[J]. Computer Science, 2015.

[7.25] Bell S, Zitnick C L, Bala K, et al. Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks[J]. 2015: 2874-2883.

[7.26] Collins J, Sohldickstein J, Sussillo D. Capacity and Trainability in Recurrent Neural Networks [J]. 2016.

[7.27] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[J]. 2015: 770-778.

[7.28] Jou B, Chang S F. Deep Cross Residual Learning for Multitask Visual Recognition[C]. ACM on Multimedia Conference. ACM, 2016: 998-1007.

[7.29] He K, Zhang X, Ren S, et al. Identity Mappings in Deep Residual Networks[J]. 2016.

[7.30] Ishikawa Y, Washiya K, Aoki K, et al. Brain tumor classification of microscopy images using deep residual learning[C]. SPIE BioPhotonics Australasia. 2016: 100132Y.

[7.31] Han Y S, Yoo J, Ye J C. Deep Residual Learning for Compressed Sensing CT Reconstruction via Persistent Homology Analysis[J]. 2016.

[7.32] Bae W, Yoo J, Ye J C. Beyond Deep Residual Learning for Image Restoration: Persistent Homology-Guided Manifold Simplification[J]. 2016.

[7.33] Yang W, Feng J, Yang J, et al. Deep Edge Guided Recurrent Residual Learning for Image Super-Resolution[J]. 2016.

[7.34] Shah A, Kadam E, Shah H, et al. Deep Residual Networks with Exponential Linear Unit[C]. International Symposium on Computer Vision and the Internet. ACM, 2016: 59-65.

[7.35] Boyd S, Parikh N, Chu E, et al. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers[J]. Foundations & Trends® in Machine Learning,

2010，3(1)：1-122.

[7.36] Y Yang，J Sun，H Li，et al. Deep ADMM-Net for Compressive Sensing MRI，2016

[7.37] Larsson G，Maire M，Shakhnarovich G. FractalNet：Ultra-Deep Neural Networks without Residuals[J]. 2017.

[7.38] Mnih V，Kavukcuoglu K，Silver D，et al. Playing Atari with Deep Reinforcement Learning[J]. Computer Science，2013.

[7.39] Mnih V，Kavukcuoglu K，Silver D，et al. Human-level control through deep reinforcement learning. [J]. Nature，2015，518(7540)：529-533.

[7.40] Narasimhan K，Kulkarni T，Barzilay R. Language Understanding for Text-based Games Using Deep Reinforcement Learning[J]. Computer Science，2015，40(4)：1-5.

[7.41] Hasselt H V，Guez A，Silver D. Deep Reinforcement Learning with Double Q-learning[J]. Computer Science，2015.

[7.42] Wang Z，Schaul T，Hessel M，et al. Dueling Network Architectures for Deep Reinforcement Learning[J]. Computer Science，2016.

[7.43] Zoph B，Le Q V. Neural Architecture Search with Reinforcement Learning[J]. 2016.

[7.44] Jaques N，Gu S，Turner R E，et al. Tuning Recurrent Neural Networks with Reinforcement Learning[J]. 2016.

[7.45] Kulkarni T D，Narasimhan K R，Saeedi A，et al. Hierarchical Deep Reinforcement Learning：Integrating Temporal Abstraction and Intrinsic Motivation[J]. 2016.

[7.46] Baker B，Gupta O，Naik N，et al. Designing Neural Network Architectures using Reinforcement Learning[J]. 2016.

[7.47] Kulkarni T D，Narasimhan K R，Saeedi A，et al. Hierarchical Deep Reinforcement Learning：Integrating Temporal Abstraction and Intrinsic Motivation[J]. 2016.

[7.48] Goodfellow I J，Pougetabadie J，Mirza M，et al. Generative Adversarial Nets[J]. Advances in Neural Information Processing Systems，2014，3：2672-2680.

[7.49] Jebara T. Discriminative，generative and imitative learning[M]. Massachusetts Institute of Technology，2002.

[7.50] Tu Z. Learning Generative Models via Discriminative Approaches[C]. Computer Vision and Pattern Recognition，2007. CVPR'07. IEEE Conference on. IEEE，2007：1-8.

[7.51] Zhu J，Ahmed A，Xing E P. MedLDA：maximum margin supervised topic models[J]. The Journal of Machine Learning Research，2012，13(1)：2237-2278.

[7.52] Nowozin S，Cseke B，Tomioka R. f-GAN：Training Generative Neural Samplers using Variational Divergence Minimization[J]. 2016.

[7.53] Zhao J，Mathieu M，Lecun Y. Energy-based Generative Adversarial Network[J]. 2017.

[7.54] Chen X，Duan Y，Houthooft R，et al. InfoGAN：Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets[J]. 2016.

[7.55] Larsen A B L，Nderby S，Ren K，et al. Autoencoding beyond pixels using a learned similarity metric[C]. International Conference on International Conference on Machine Learning. JMLR. org，2016：1558-1566.

[7.56] Springenberg J T. Unsupervised and Semi-supervised Learning with Categorical Generative

Adversarial Networks[J]. Computer Science, 2015.
[7.57] Dosovitskiy A, Brox T. Generating Images with Perceptual Similarity Metrics based on Deep Networks[J]. 2016.
[7.58] Johnson J, Alahi A, Li F F. Perceptual Losses for Real-Time Style Transfer and Super-Resolution [J]. 2016.
[7.59] Ledig C, Theis L, Huszar F, et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network[J]. 2016.
[7.60] Reed S, Akata Z, Yan X, et al. Generative Adversarial Text to Image Synthesis[J]. 2016.
[7.61] Vondrick C, Pirsiavash H, Torralba A. Generating Videos with Scene Dynamics[J]. 2016.
[7.62] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks[J]. Computer Science, 2015.
[7.63] Salimans T, Goodfellow I, Zaremba W, et al. Improved Techniques for Training GANs [J]. 2016.
[7.64] Qi G J. Loss-Sensitive Generative Adversarial Networks on Lipschitz Densities[J]. arXiv preprint arXiv: 1701.06264, 2017.
[7.65] Li J, Monroe W, Shi T, et al. Adversarial Learning for Neural Dialogue Generation[J]. 2017.
[7.66] Sønderby C K, Caballero J, Theis L, et al. Amortised MAP Inference for Image Super-resolution [J]. 2017.
[7.67] Ravanbakhsh S, Lanusse F, Mandelbaum R, et al. Enabling Dark Energy Science with Deep Generative Models of Galaxy Images[J]. 2016.
[7.68] Ho J, Ermon S. Generative Adversarial Imitation Learning[J]. 2016.
[7.69] Zhu J Y, Krähenbühl P, Shechtman E, et al. Generative Visual Manipulation on the Natural Image Manifold[J]. 浙江大学学报(英文版)(c辑：计算机与电子), 2016, 15(4): 312-320.
[7.70] Isola P, Zhu J Y, Zhou T, et al. Image-to-Image Translation with Conditional Adversarial Networks[J]. 2016.
[7.71] Shrivastava A, Pfister T, Tuzel O, et al. Learning from Simulated and Unsupervised Images through Adversarial Training[J]. 2016.
[7.72] Ledig C, Theis L, Huszar F, et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network[J]. 2016.
[7.73] Nguyen A, Yosinski J, Bengio Y, et al. Plug & Play Generative Networks: Conditional Iterative Generation of Images in Latent Space[J]. 2016.
[7.74] Yu L, Zhang W, Wang J, et al. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient[J]. 2016.
[7.75] Lotter W, Kreiman G, Cox D. Unsupervised Learning of Visual Structure using Predictive Generative Networks[J]. Computer Science, 2015.
[7.76] Reed S, Akata Z, Yan X, et al. Generative Adversarial Text to Image Synthesis[J]. 2016.
[7.77] Brock A, Lim T, Ritchie J M, et al. Neural Photo Editing with Introspective Adversarial Networks[J]. 2016.
[7.78] Zhang H, Xu T, Li H, et al. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks[J]. 2016.

[7.79] Higgins I, Matthey L, Glorot X, et al. Early Visual Concept Learning with Unsupervised Deep Learning[J]. 2016.

[7.80] Radford A, Metz L, Chintala S. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks[J]. Computer Science, 2015.

[7.81] Nowozin S, Cseke B, Tomioka R. f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization[J]. 2016.

[7.82] Zili Yi, Hao Zhang, Ping Tan. Minglun Gong. DualGAN: Unsupervised Dual Learning for Image-to-Image Translation[J]. e-Print arXiv 2017.

第8章



深度复卷积神经网络与深度二值神经网络

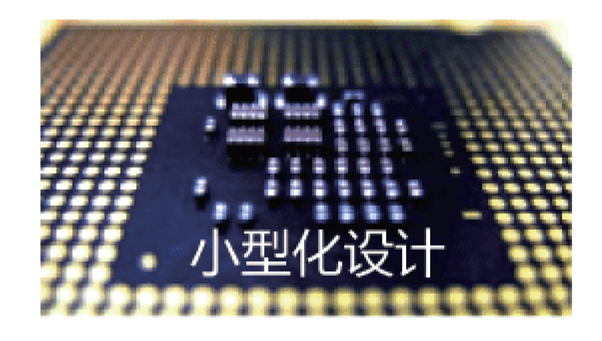
CHAPTER 8

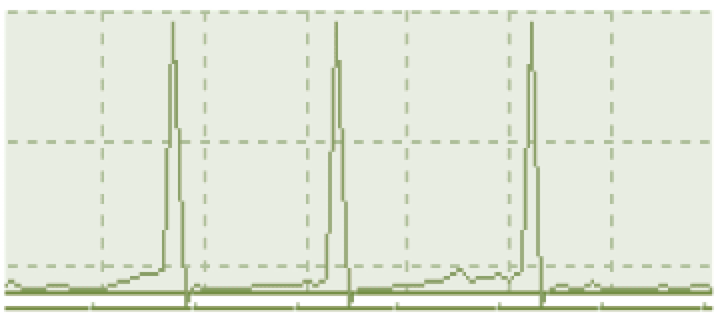
扩张深度神经网络——数域延拓、简化约束

■ 深度复卷积神经网络——复数域ℂ

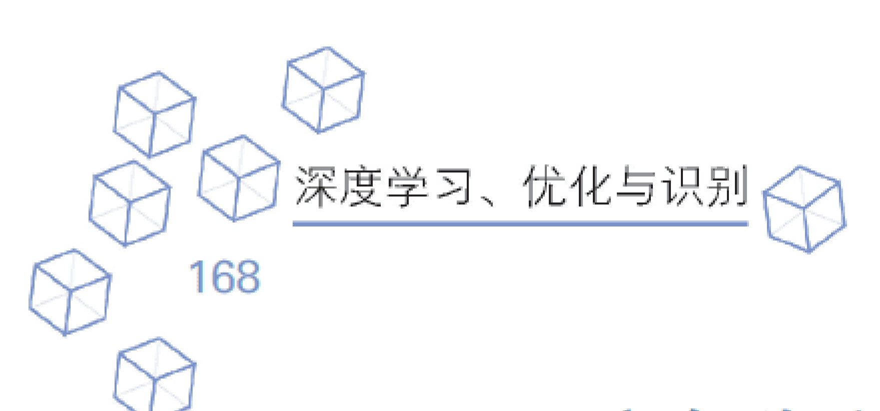
■ 深度二值神经网络——二值化、小而快的神经网络时代

■ 深度脉冲神经网络——生物神经脉冲响应





脉冲响应



8.1 深度复卷积神经网络

8.1.1 网络模型构造的动机

关于深度卷积神经网络已在第3章详细介绍过，由于实际应用中数据所在的域通常为实数域$\mathbb{R}$，所以常见的深度卷积神经网络的计算环境也是实数域；若对于复数域$\mathbb{C}$上的数据，例如极化SAR影像分类任务，其处理流程图如图8.1所示。

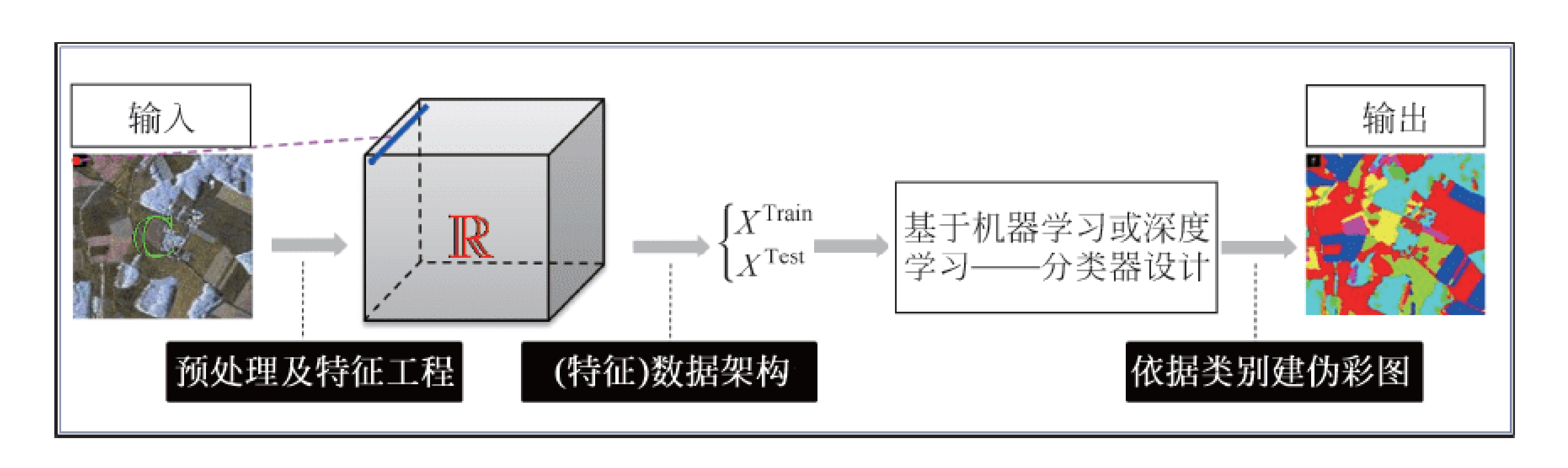


图8.1 极化SAR影像分类流程图

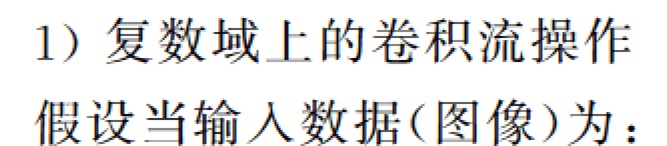
图8.1中预处理及特征工程包含滤波增强等处理，以及根据物理与统计特性进行特征提取与筛选(如降维处理的流形降维、PCA降维等)，处理完后便可以建立复数域(即空域)到实数域(即特征域)的"映射"，即输入中的每一个位置或"像素"可由特征域上的一向量进行描述或刻画，通常这一步处理的好坏取决于对(空域)数据先验的认知；随后的(特征)数据架构便基于特征域来实现"逐像素"特征整理——训练与测试集，进一步，基于机器学习(如SVM、神经网络等)或深度学习(如深度置信网络、深度卷积神经网络)实现分类；最后，依据(逐像素)分类结果实现伪彩图输出。可问题是这种处理方式将数据从复数域到实数域的转化过程中，是否有信息损失？损失是否影响网络的性能？能否直接在复数域上对数据进行处理，以削弱对数据中某种先验(物理及统计特性)的依赖性？基于此，本节将对深度复卷积神经网络给出积极的探索。

8.1.2 网络模型的数学物理描述

本节仅探索实数域上的深度卷积神经网络向复数域进行延拓时，相应的操作如何进行？以及整个深度复卷积神经网络的结构和数学理解。

1. 模块延拓的数学物理描述

深度卷积神经网络主要包含卷积流和全连接层模块，下面将详述这两个模块从实数域向复数域的延拓中所进行的具体操作。

1）复数域上的卷积流操作

假设当输入数据（图像）为：

$$\boldsymbol{x} = \boldsymbol{x}_{\mathrm{Re}} + \mathrm{j} \cdot \boldsymbol{x}_{\mathrm{Im}} \in \mathbb{C}^{n \times m} \tag{8.1}$$

其中符号 j 为虚数单位。

（1）卷积操作——放大或缩小

假设卷积核或滤波器为：

$$\boldsymbol{w} = \boldsymbol{w}_{\mathrm{Re}} + \mathrm{j} \cdot \boldsymbol{w}_{\mathrm{Im}} \in \mathbb{C}^{u \times v} \tag{8.2}$$

则关于输入进行卷积操作后有：

$$\boldsymbol{x} * \boldsymbol{w} = (\boldsymbol{x}_{\mathrm{Re}} * \boldsymbol{w}_{\mathrm{Re}} - \boldsymbol{x}_{\mathrm{Im}} * \boldsymbol{w}_{\mathrm{Im}}) + \mathrm{j} \cdot (\boldsymbol{x}_{\mathrm{Im}} * \boldsymbol{w}_{\mathrm{Re}} + \boldsymbol{x}_{\mathrm{Re}} * \boldsymbol{w}_{\mathrm{Im}}) \in \mathbb{C}^{n_1 \times m_1} \tag{8.3}$$

其中“ * ”仍为卷积的操作，其操作的属性仍分为 Full 卷积、Same 卷积和 Valid 卷积等。不同属性下的卷积所对应的尺寸大小不一致。不失一般性，这里描述的卷积属性为 Valid，则有：

$$\begin{cases} n_1 = n - u + 1 \\ m_1 = m - v + 1 \end{cases} \tag{8.4}$$

更多关于卷积操作的描述请参考第 3 章深度卷积神经网络，另外关于实数域上的反卷积及转置卷积等概念可相应地移植到复数域上。通常，卷积操作中有两个参数，一个是步长 Stride，另一个是覆盖 Padding 的方式（Full Padding 和 Zero Padding）与尺寸。在 Valid 卷积下仍使用如下公式计算特征图（每一维度）的新尺寸：

$$\text{newsize} = \left\lfloor \frac{\text{inputsize} - \text{kernelsize} + 2 \cdot \text{padding}}{\text{stride}} \right\rfloor + 1 \tag{8.5}$$

其中的 kernelsize 为滤波器的尺寸，inputsize 为输入的尺寸，注意式（8.4）中的 Stride 为 1，Zero Padding 为 0。

（2）池化操作——平移不变性

假设卷积操作后的特征图记为：

$$\boldsymbol{f} = \boldsymbol{x} * \boldsymbol{w} + \eta \in \mathbb{C}^{n_1 \times m_1} \tag{8.6}$$

注意这里的 η 为偏置，即 $\eta = \eta_{\mathrm{Re}} + \mathrm{j} \cdot \eta_{\mathrm{Im}} \in \mathbb{C}$，复数矩阵与复数相加，即矩阵中的每一个元素与该复数相加。进一步，假设池化操作的半径记为 r，则对特征图 $\boldsymbol{f}$ 进行池化后得：

$$P(\boldsymbol{f}) = P_{\mathrm{Re}}(\boldsymbol{f}) + \mathrm{j} \cdot P_{\mathrm{Im}}(\boldsymbol{f}) \in \mathbb{C}^{n_2 \times m_2} \tag{8.7}$$

其中 P 为池化（pooling）操作，则有：

$$\begin{cases} n_2 = \left\lfloor \dfrac{n_1}{r} \right\rfloor \\ m_2 = \left\lfloor \dfrac{m_1}{r} \right\rfloor \end{cases} \tag{8.8}$$

操作 $\lfloor \cdot \rfloor$ 为向下取整。

备注：这里的池化操作也可以为空域塔式池化。

（3）非线性（激活）操作——扭曲特性

实数域上的激活函数通过如下方式延拓至复数域上，即



$$\boldsymbol{T}=\sigma(P(\boldsymbol{f}))\triangleq\sigma(P_{\mathrm{Re}}(\boldsymbol{f}))+\mathrm{j}\cdot\sigma(P_{\mathrm{Im}}(\boldsymbol{f}))\in\mathbb{C}^{n_2\times m_2} \tag{8.9}$$

其中的 $\sigma(\cdot)$为非线性激活函数，常用的有修正线性单元 ReLU。

（4）批量归一化（局部响应归一化）操作——加速计算并保持拓扑结构对应性。

归一化操作与实数域一致，即

$$\boldsymbol{K}=N(\boldsymbol{T})\triangleq N(\boldsymbol{T}_{\mathrm{Re}})+\mathrm{j}\cdot N(\boldsymbol{T}_{\mathrm{Im}})\in\mathbb{C}^{n_2\times m_2} \tag{8.10}$$

这里的 $N(\cdot)$为归一化操作（或局部响应归一化）。

2）复数域上的全连接层操作

当获取若干卷积流处理后的特征图（Feature Maps）后，通常会通过拉伸或向量化操作得到相应的特征，再通过全连接层进一步处理，即

$$\begin{cases}\boldsymbol{V}=\mathrm{vec}(\boldsymbol{K})\in\mathbb{C}^{n_3}\\ \boldsymbol{f}_{\mathrm{FC}}=\sigma(\boldsymbol{W}\cdot\boldsymbol{V}+\boldsymbol{b})\in\mathbb{C}^{n_4}\\ \boldsymbol{W}\in\mathbb{C}^{n_4\times n_3},\boldsymbol{b}\in\mathbb{C}^{n_4}\end{cases} \tag{8.11}$$

其中的 vec$(\cdot)$为矩阵向量化的操作，即通过按行或按列，抑或按“之”字形进行拉升。另外这里的 $\sigma(\cdot)$为非线性函数，与式(8.9)的计算一致；另外矩阵与向量相乘，相加操作符合复数域上的运算法则。

2. 模型的数学理解

下面仿照实数域上的 LeNet5 网络，给出复数域上网络模型的结构（图 8.2）并详细地分析如下。

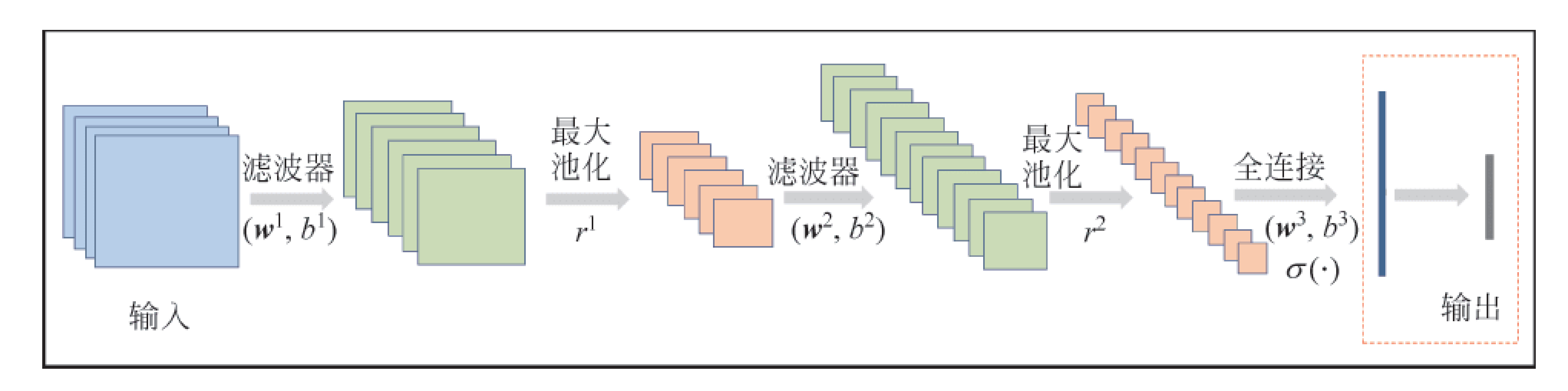


图 8.2　深度复卷积网络——复数域上 LeNet5 网络

1）数据（训练数据集）

$$\{\boldsymbol{x}^{(t)},\boldsymbol{y}^{(t)}\}_{t=1}^{N} \tag{8.12}$$

其中输入 $\boldsymbol{x}^{(t)}$ 为：

$$\boldsymbol{x}^{(t)}\in\mathbb{C}^{n\times m@\kappa} \tag{8.13}$$

即 κ 个通道，每个通道下的尺寸为 $n\times m$；另外 $\boldsymbol{y}^{(t)}$ 为对应的类标。

2）模型

根据图 8.2，假设输入为 $\boldsymbol{x}\in\mathbb{C}^{n\times m@\kappa}$，前向传输至第二个全连接层的输出记为 $\boldsymbol{X}\in\mathbb{C}^{s}$（也称为深层抽象特征），即利用如下的公式：



$$\begin{cases} \boldsymbol{X} = \varphi(\boldsymbol{x}, \vartheta) \\ \vartheta = [\boldsymbol{w}^1, \boldsymbol{w}^2, \boldsymbol{w}^3; \boldsymbol{b}^1, \boldsymbol{b}^2, \boldsymbol{b}^3] \end{cases} \tag{8.14}$$

其中的$(\boldsymbol{w}^i, \boldsymbol{b}^i)_{i=1}^3$为相应层级复数域上的滤波器与偏置。

获取特征表示 $\boldsymbol{X} \in \mathbb{C}^s$ 后，针对分类任务，如何设计分类器（即图 8.2 中虚线框的理解）？下面给出两种设计分类器的方式，第一种是将数据 $\boldsymbol{x}$ 所对应的类标 $\boldsymbol{y}$ 扩展为复数域，即如分类类别个数为 C，且 $\boldsymbol{x}$ 对应的类标为第二类，则输出 $\boldsymbol{y}$ 可写为：

$$\boldsymbol{y} = \begin{pmatrix} 0 + \mathrm{j} \cdot 0 \\ 1 + \mathrm{j} \cdot 1 \\ \vdots \\ 0 + \mathrm{j} \cdot 0 \end{pmatrix} \in \mathbb{C}^C \tag{8.15}$$

设计的（非线性）分类器有：

$$\boldsymbol{y} = \sigma(\boldsymbol{W} \cdot \boldsymbol{X} + \boldsymbol{\beta}) \tag{8.16}$$

其中参数：

$$\begin{cases} \boldsymbol{W} \in \mathbb{C}^{C \times s} \\ \boldsymbol{\beta} \in \mathbb{C}^C \end{cases} \tag{8.17}$$

备注：式(8.16)中的分类器也可包含复数域上 Softmax，即

$$\begin{cases} y(c) = P(\mathrm{label}(\boldsymbol{x}) = c \mid \boldsymbol{X}, \boldsymbol{\theta}_c) = \dfrac{\mathrm{e}^{\boldsymbol{X}^{\mathrm{T}} \cdot \boldsymbol{\theta}_c}}{\sum_{j=1}^{C} \mathrm{e}^{\boldsymbol{X}^{\mathrm{T}} \cdot \boldsymbol{\theta}_j}} \in \mathbb{C} \\ \boldsymbol{\theta}_c \in \mathbb{C}^s, c = 1, 2, \cdots, C \end{cases} \tag{8.18}$$

其中的 $y(c)$表示输出 $\boldsymbol{y} \in \mathbb{C}^C$ 的第 c 个元素，待学习的参数为$(\boldsymbol{\theta}_c)_{c=1}^C$。

第二种分类器的设计是直接将输入的深层抽象特征 $\boldsymbol{X} \in \mathbb{C}^s$，转化为实数域上的特征，例如将其实部与虚部堆栈形成：

$$\widetilde{\boldsymbol{X}} = \begin{pmatrix} \boldsymbol{X}_{\mathrm{Re}} \\ \boldsymbol{X}_{\mathrm{Im}} \end{pmatrix} \in \mathbb{R}^{2 \cdot s} \tag{8.19}$$

或者利用其模与幅角（辐角主值）构成实数域上的特征并去量纲归一化。根据得到的实数域上的特征进行熟悉的分类器设计，注意此时的输出 $\boldsymbol{y}$ 不用扩展至复数域。

3）优化目标函数

关于分类器的设计采纳第一种设计思路，利用式(8.14)和式(8.18)建立的输入与输出之间的关系，得到优化目标函数为：

$$\begin{cases} \min\limits_{\vartheta, \theta} J(\vartheta, \theta) = -\dfrac{1}{T} \sum_{t=1}^{T} \sum_{c=1}^{C} \delta(\mathrm{label}(\boldsymbol{x}^{(t)}) = c) \cdot [\log(\hat{\boldsymbol{y}}_{\mathrm{Re}}^{(t)}(c)) + \log(\hat{\boldsymbol{y}}_{\mathrm{Im}}^{(t)}(c))] \\ \boldsymbol{\theta} = \{\boldsymbol{\theta}_c\}_{c=1}^C \end{cases} \tag{8.20}$$

其中 $\delta(\cdot)$为狄利克雷函数或示性函数，另外$\hat{\boldsymbol{y}}_{\mathrm{Re}}^{(t)}(c)$为第 t 个样本预测为第 c 类的概率的实



部，类似$\hat{\boldsymbol{y}}_{\mathrm{Im}}^{(t)}(c)$为相应的虚部；待优化的参数包括特征学习阶段的$\vartheta$和分类器设计阶段的$\boldsymbol{\theta}$。

对于回归任务而言，可利用能量损失来建立如下的优化目标函数（依据两个复数相等，即实部与虚部所对应的值相同）：

$$\begin{cases}\min\limits_{\vartheta,\boldsymbol{W},\boldsymbol{\beta}} J(\vartheta,\boldsymbol{W},\boldsymbol{\beta}) = \dfrac{1}{T}\sum\limits_{t=1}^{T}[\parallel \hat{\boldsymbol{y}}_{\mathrm{Re}}^{(t)} - \boldsymbol{y}_{\mathrm{Re}}^{(t)} \parallel_2^2 + \parallel \hat{\boldsymbol{y}}_{\mathrm{Im}}^{(t)} - \boldsymbol{y}_{\mathrm{Im}}^{(t)} \parallel_2^2] \\ \hat{\boldsymbol{y}}^{(t)} = \sigma(\boldsymbol{W}\cdot\boldsymbol{X}^{(t)}+\boldsymbol{\beta}) = \sigma(\boldsymbol{W}\cdot\varphi(\boldsymbol{x}^{(t)},\vartheta)+\boldsymbol{\beta})\end{cases} \tag{8.21}$$

4）求解

关于优化目标函数(8.20)的求解，可以利用随机梯度下降的方式来实现，即分类器设计阶段的更新公式为：

$$\begin{cases}\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha\cdot ?\boldsymbol{\theta}\,|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(k)}} \\ ?\boldsymbol{\theta}_c = \dfrac{\partial J(\vartheta,\boldsymbol{\theta})}{\partial\,\boldsymbol{\theta}_c} \\ ?\boldsymbol{\theta} = (?\boldsymbol{\theta}_1, ?\boldsymbol{\theta}_2, \cdots, ?\boldsymbol{\theta}_C)\end{cases} \tag{8.22}$$

特征学习阶段的更新公式为：

$$\begin{cases}\vartheta^{(k+1)} = \vartheta^{(k)} - \alpha\cdot ?\vartheta\,|_{\vartheta=\vartheta^{(k)}} \\ ?\vartheta = \dfrac{\partial J(\vartheta,\boldsymbol{\theta})}{\partial\vartheta}\end{cases} \tag{8.23}$$

其中的α为学习速率。求解过程中核心计算误差传播项，复数域可分为实部与虚部分别计算。

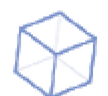
备注：关于深度复卷积神经网络的应用，如极化 SAR 图像的分类任务和实验技巧可参考第 12 章。

8.2 深度二值神经网络

8.2.1 网络基本结构

传统的深度神经网络通常关于前向传播和反向传播使用的是浮点计算，然而该计算所需大的存储空间和大的计算量，严重地阻碍其（如便携设备）应用；为此，人们探索了大量的二值化或三值化等策略，期望降低对存储空间和计算量的需求以达到便携的应用目的，但可惜的是一直以来难以达到高预测准确率。最新的研究进展给出，加拿大蒙特利尔大学 Yoshua Bengio 团队提出的二值化神经网络设法让计算主要在-1和$+1$之间进行，可以成几十倍地降低内存和计算量并使预测准确率达到实用水平，但仍需指出的是二值神经网络的信息损失相对于浮点精度下的神经网络是非常大的。

备注：二值（化）神经网络模型与输入输出限制在(0,1)之间的受限玻尔兹曼机和二元域上逻辑操作的神经网络不同。



8.2.2　网络的数学物理描述

本节给出的深度二值神经网络是 Yoshua Bengio 团队提出的，下面详细地给出论文的数学分析。

注意该网络并没有对输入也要求是二值化的，另外针对分类任务，输出为类标是不变的，但该模型对于回归任务(逼近)的性能尚未研究，通常二值神经网络随着层级的增加，信息的损失相对于浮点计算的神经网络是非常大的。注意这里针对分类任务进行如下分析，假设类别个数为 C。

1. 数据(训练数据集)

$$\{\boldsymbol{x}^{(n)} \in \mathbb{R}^m, \boldsymbol{y}^{(n)} \in \{0,1\}^C\}_{n=1}^N \tag{8.24}$$

其中 $\boldsymbol{y}^{(n)}$ 为输入 $\boldsymbol{x}^{(n)}$ 所对应的输出(类标指示)。

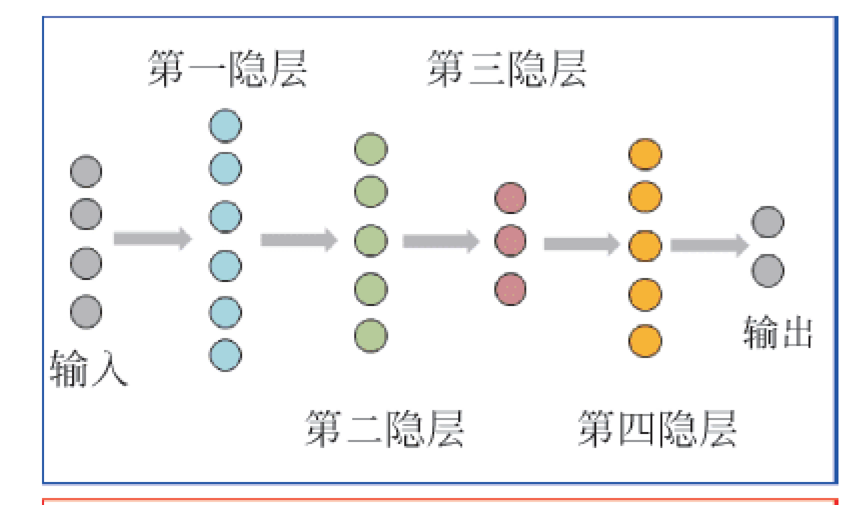
2. 模型

首先给出经典的深度前馈神经网络关于层级状态 $\boldsymbol{s}_l$ 与响应或激活值 $\boldsymbol{a}_l$ 的关系式：

$$\begin{cases} \boldsymbol{s}_l = \boldsymbol{W}_l \cdot \boldsymbol{f}_{l-1} + \boldsymbol{b}_l \\ \boldsymbol{f}_l = \sigma^{(l)}(\boldsymbol{s}_l) \end{cases} \tag{8.25}$$

其中 $l=1,2,\cdots,L$，$\sigma^{(l)}(\cdot)$为第 l 隐层上的激活函数。接下来，依据图 8.3 中的要求，将其改为二值化的传播形式：

$$\begin{cases} \boldsymbol{s}_l = \boldsymbol{W}_l \cdot \boldsymbol{f}_{l-1} \\ \boldsymbol{f}_l = \sigma^{(l)}(\boldsymbol{s}_l) \\ \boldsymbol{W}_l \in \{-1,1\}^{m_l \times m_{l-1}}; \boldsymbol{f}_l \in \{-1,1\}^{m_l} \end{cases} \tag{8.26}$$



要求 $\begin{cases} W_l \text{ 每一层权值矩阵中的元素是二值的} \\ f_l \text{ 每一层激活值的元素是二值的} \end{cases}$

其中l=1,2,…,L

图 8.3　深度二值神经网络的要求

其中 $m_l(l=1,2,\cdots,L)$为第 l 个隐层上的激活单元个数，另外 $\sigma^{(l)}(\cdot)$为非线性激活操作，符号$\{-1,1\}^{u\times v}$为二值化的 $u\times v$ 维矩阵，注意式(8.26)与式(8.25)的区别。

通常，实际操作中，会对每一隐层的激活值进行批量归一化操作，即在数据经过一层进入下一层之前，使之均值为0，方差为1。这样处理的优点包括：一是加速训练过程；二是减少权重的值的尺度的影响；三是批量归一化所带来的噪声具有模型正则化的作用(本质上为乘性与加性噪声的叠加)。例如，假设数据集(8.24)分为 P 批，每一批数量为 M，则有：

$$N = P \cdot M \tag{8.27}$$

对于每一批数据，假设得到第 l 隐层上的激活值，记为：

$$\{\boldsymbol{x}^{(\kappa)}\}_{\kappa=1}^M \longrightarrow \{\boldsymbol{f}_l^{(\kappa)}\}_{\kappa=1}^M \tag{8.28}$$

则批量归一化操作，即对激活值，依据其均值与方差，实现归一化，但考虑到二值化网络的特性，使用的归一化操作为 Shift based Batch Normalization Transform，即写为：

$$\{\tilde{\boldsymbol{f}}_l^{(\kappa)}\}_{\kappa=1}^{M} = \text{BatchNorm}(\{\boldsymbol{f}_l^{(\kappa)}\}_{\kappa=1}^{M}, \boldsymbol{\theta}_l) \tag{8.29}$$

其中的$\boldsymbol{\theta}_l$ 为第 l 隐层的待学习参数，注意这里仍使用均值和方差来实现归一化操作，篇幅与理解所限，具体请结合代码和论文的分析。

最后，具体地给出$\{\boldsymbol{f}_l, \boldsymbol{W}_l\}_{l=1}^{L}$是如何进行二值化的。再强调一点：该网络的二值化仅限于权值矩阵与每一层上的激活值(除输入层外)。

$$\begin{cases} \boldsymbol{f}_l^b = \text{Binarize}(\boldsymbol{f}_l) \\ \boldsymbol{W}_l^b = \text{Binarize}(\boldsymbol{W}_l) \end{cases} \tag{8.30}$$

常用的二值化函数 Binarize(·)包括确定性和随机性两种，确定性的为：

$$\nu^b = \text{sign}(v) = \begin{cases} +1 & \text{if } \nu \geqslant 0 \\ -1 & \text{else} \end{cases} \tag{8.31}$$

随机性的为：

$$\nu^b = \begin{cases} +1 & \text{with probability } p = \zeta(\nu) \\ -1 & \text{with probability } 1 - p \end{cases} \tag{8.32}$$

其中的

$$\zeta(\nu) = \max\left(0, \min\left(1, \frac{1+\nu}{2}\right)\right) \tag{8.33}$$

注意式(8.31)、式(8.32)和式(8.33)是逐元素进行的，可推广至式(8.30)。

3. 优化目标函数

依据式(8.26)，其中输入数据 $\boldsymbol{x}$ 记为：

$$\boldsymbol{f}_0 = \boldsymbol{x} \tag{8.34}$$

但是这里的 $\boldsymbol{f}_0$ 不为二值化。进一步，假设对于输入 $\boldsymbol{x}$ 得到第 L 隐层的二值化输出为 $\boldsymbol{f}_L^b$(为抽象特征)，后面的分类器可以利用支撑向量机或 Softmax 等，不失一般性，非线性分类器可以写为：

$$\boldsymbol{y} = \sigma^{(L+1)}(\boldsymbol{W}_{L+1} \cdot \boldsymbol{f}_L^b) \tag{8.35}$$

若进一步要求 $\boldsymbol{W}_{L+1}$是二值化的，则可以将式(8.26)和式(8.35)统一起来，只需将层级数变为 $L+1$(包含一个输出层)。

优化目标函数为：

$$\min_{(\boldsymbol{W},\boldsymbol{\theta})} J(\boldsymbol{W}, \boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N} \text{loss}(\hat{\boldsymbol{y}}^{(n)}, \boldsymbol{y}^{(n)}) + \lambda \cdot R(\boldsymbol{W}) \tag{8.36}$$

其中的 $\boldsymbol{W}$ 和$\boldsymbol{\theta}$ 分别为权值矩阵和批量归一化操作中的参数，即有：

$$\begin{cases} \boldsymbol{W} = \{\boldsymbol{W}_l\}_{l=1}^{L+1} \\ \boldsymbol{\theta} = \{\boldsymbol{\theta}_l\}_{l=1}^{L} \end{cases} \tag{8.37}$$




注意输出层不需要批量归一化。另外，损失函数 loss(・)可以使用交互熵的形式来构造，符号 $R(\boldsymbol{W})$ 为关于权值矩阵 $\boldsymbol{W}$ 的正则性约束。

4. 求解

关于优化目标函数(8.36)，给出求解框架之前，先给出几个先决或假设条件：

假设一：超参数(包括层数、隐单元个数和激活函数)等已给定。

假设二：使用随机梯度下降的方法，且利用式(8.27)对数据进行分批次处理。

假设三：二值化函数使用确定性的，即式(8.31)。

假设四：关于参数，即$(\boldsymbol{W},\boldsymbol{\theta})$，随机初始化，且对于权值矩阵实现二值化处理。

基于以上的假设条件，已知：

$$\{\boldsymbol{W}^{(0)},\boldsymbol{\theta}^{(0)}\}=\{\{\boldsymbol{W}_l^{(0)}\}_{l=1}^{L+1},\{\boldsymbol{\theta}_l^{(0)}\}_{l=1}^{L}\} \tag{8.38}$$

其中"(0)"代表初始化参数，假设迭代次数为 T，得到：

$$\{\boldsymbol{W}^{(T)},\boldsymbol{\theta}^{(T)}\} \tag{8.39}$$

并期望达到：

$$\begin{cases} \|\boldsymbol{W}^{(T)}-\boldsymbol{W}^*\|_2^2\leqslant\xi \\ \|\boldsymbol{\theta}^{(T)}-\boldsymbol{\theta}^*\|_2^2\leqslant\xi \end{cases} \tag{8.40}$$

其中 ξ 为指定误差，$\boldsymbol{W}^*$ 和$\boldsymbol{\theta}^*$ 为式(8.36)的解；通常，可以可视化这一期望的方式是绘制目标函数(8.36)随着迭代次数变化的值，即

$$J(\boldsymbol{W}^{(t)},\boldsymbol{\theta}^{(t)})(t=1,2,\cdots,T) \tag{8.41}$$

并满足该值的趋势是整体下降，允许局部有所上升(震荡)。

针对某一次迭代，如已知 t 次的参数值，即

$$\{\boldsymbol{W}^{(t)},\boldsymbol{\theta}^{(t)}\} \tag{8.42}$$

如何更新 $t+1$ 次的参数值？由于对数据进行分批处理，即批量 Epoch$=P$，且每一批的数据为 $N_p=M$ 且 $p=1,2,\cdots,P$。

备注：这里是对数据集(8.24)进行"平均化"划分，即每一批数据相同，也可随机式划分或随着批次示数 p 的增加，批次数据呈某种"趋势"变换(如下降)。

先根据 $p=1$ 时，即 $N_1=M$ 个数据来更新第 t 次的参数值，得到：

$$(\boldsymbol{W}^{(t+1,1)},\boldsymbol{\theta}^{(t+1,1)})=\text{Update}(\boldsymbol{W}^{(t)},\boldsymbol{\theta}^{(t)},N_1) \tag{8.43}$$

其中 Update(・)包含三步，一是目标函数关于参数的偏导数，并计算梯度的下降量；二是给定学习速率计算更新后的参数；三是对参数中的权值矩阵(每一层上的)进行二值化处理。下面给出这三步的数学分析。

1) 第一步：计算偏导数与梯度下降量

依据优化目标函数(8.36)，由于此时的数据量仅为 $N_1=M$ 个数据，所以：

$$\min_{(\boldsymbol{W},\boldsymbol{\theta})}J_1(\boldsymbol{W},\boldsymbol{\theta})=\frac{1}{N_1}\sum_{n=1}^{N_1}\text{loss}(\hat{\boldsymbol{y}}^{(n)},\boldsymbol{y}^{(n)})+\lambda\cdot R(\boldsymbol{W}) \tag{8.44}$$

其中 $J_1(\boldsymbol{W},\boldsymbol{\theta})$中的角标“1”为 $p=1$ 的意思；并计算

$$\begin{cases}\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{W}}=\left[\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{W}_1},\cdots,\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{W}_{L+1}}\right]\\ \dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}=\left[\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_1},\cdots,\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_L}\right]\end{cases}\tag{8.45}$$

在反向传播算法中，逐层偏导数的求解核心依赖逐层误差传播项的建立和链式法则的应用。进一步，可得梯度下降量为：

备注：这部分为整个求解算法的核心，篇幅与理解所限，请结合源代码和参考文献进一步理解，这里仅给出框架。

$$\begin{cases}?\boldsymbol{W}\big|_{\boldsymbol{W}=\boldsymbol{W}^{(t)}}=\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{W}}\bigg|_{\boldsymbol{W}=\boldsymbol{W}^{(t)}}=\left[\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{W}_1}\bigg|_{\boldsymbol{W}_1=\boldsymbol{W}_1^{(t)}},\cdots,\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{W}_{L+1}}\bigg|_{\boldsymbol{W}_{L+1}=\boldsymbol{W}_{L+1}^{(t)}}\right]\\ ?\boldsymbol{\theta}\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t)}}=\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t)}}=\left[\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_1}\bigg|_{\boldsymbol{\theta}_1=\boldsymbol{\theta}_1^{(t)}},\cdots,\dfrac{\partial J_1(\boldsymbol{W},\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_L}\bigg|_{\boldsymbol{\theta}_L=\boldsymbol{\theta}_L^{(t)}}\right]\end{cases}\tag{8.46}$$

注意参数下面的指标为层级示数，例如 $\boldsymbol{W}_1$，$\boldsymbol{\theta}_1$ 角标“1”为相应层级上的参数。

2）第二步：更新参数

第 $t+1$ 次所有批量下(即随着 $p=1,2,\cdots,P$ 的变化)的参数更新所使用的学习速率记为 $\alpha^{(t+1)}$，所以参数更新为：

$$\begin{cases}\boldsymbol{W}^{(t+1,1)}=\boldsymbol{W}^{(t)}-\alpha^{(t+1)}\cdot ?\boldsymbol{W}\big|_{\boldsymbol{W}=\boldsymbol{W}^{(t)}}\\ \boldsymbol{\theta}^{(t+1,1)}=\boldsymbol{\theta}^{(t)}-\alpha^{(t+1)}\cdot ?\boldsymbol{\theta}\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}^{(t)}}\end{cases}\tag{8.47}$$

注意：更新并不能保证所得到的权值矩阵是二值化的。

3）第三步：二值化权值矩阵

$$\begin{cases}\boldsymbol{W}^{(t+1,1)}=\mathrm{Binarize}(\boldsymbol{W}^{(t+1,1)})=\{\mathrm{Binarize}(\boldsymbol{W}_l^{(t+1,1)})\}_{l=1}^{L+1}\\ \boldsymbol{\theta}^{(t+1,1)}=\mathrm{Binarize}(\boldsymbol{\theta}^{(t+1,1)})=\{\mathrm{Binarize}(\boldsymbol{\theta}_l^{(t,1)})\}_{l=1}^{L}\end{cases}\tag{8.48}$$

其中的二值化函数使用确定性的。

完成第 $t+1$ 次迭代中，批量 $p=1$ 的更新后，依次类推，即更新公式如下：

$$(\boldsymbol{W}^{(t+p+1,1)},\boldsymbol{\theta}^{(t+p+1,1)})=\mathrm{Update}(\boldsymbol{W}^{(t+p,1)},\boldsymbol{\theta}^{(t+p,1)},N_1)\tag{8.49}$$

其中的 $p=1,2,\cdots,P-1$。并记：

$$\begin{cases}\boldsymbol{W}^{(t+1)}=\boldsymbol{W}^{(t+1,P)}\\ \boldsymbol{\theta}^{(t+1)}=\boldsymbol{\theta}^{(t+1,P)}\end{cases}\tag{8.50}$$

这样便完成了第 $t+1$ 次迭代的更新。

8.2.3 讨论

深度二值化神经网络是网络“小型化”探索的一个重要方向，它将允许把此前只能在服务器上运行的深度神经网络转嫁在智能手表上运行，比如 VGG-16 网络。而且其优化有着

严格的数学推导与更新算法，另外该网络的核心缺陷在于随着层级的前向传播，隐层拓扑结构信息损失相对传统神经网络较大。针对深度二值神经网络的缺陷，经过数学上的理论论证，提出三值神经网络模型(Ternary Weight Networks)*，它继承了二值神经网络的优点同时模型的表达能力上大大提升，另外各方面指标超过了实际应用的水平。

参考文献

[8.1] Bruna J, Chintala S, Lecun Y, et al. A theoretical argument for complex-valued convolutional networks[J]. 2015.

[8.2] Bruna J, Mallat S. Invariant Scattering Convolution Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2013, 35(8): 1872-1886.

[8.3] Cicconet M, Geiger D, Werman M. Complex-Valued Hough Transforms for Circles[J]. Computer Science, 2015.

[8.4] Cohen N, Sharir O, Shashua A. On the Expressive Power of Deep Learning: A Tensor Analysis [J]. Computer Science, 2015.

[8.5] Georgiou G M, Koutsougeras C. Complex domain backpropagation[J]. IEEE Transactions on Circuits & Systems II Analog & Digital Signal Processing, 1992, 39(5): 330-334.

[8.6] Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation[C]. Computer Vision and Pattern Recognition. IEEE, 2013: 580-587.

[8.7] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks[J]. Journal of Machine Learning Research, 2010, 9: 249-256.

[8.8] Goodfellow I J, Wardefarley D, Mirza M, et al. Maxout Networks[J]. Computer Science, 2013: 1319-1327.

[8.9] Hebb D O. The Organization of Behaviour[J]. Journal of Applied Behavior Analysis, 1949, 25 (3): 575-577.

[8.10] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting[J]. Journal of Machine Learning Research, 2014, 15(1): 1929-1958.

[8.11] Ji S, Yang M, Yu K. 3D Convolutional Neural Networks for Human Action Recognition[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2013, 35(1): 221-231.

[8.12] Kim T, Adali T. Fully Complex Multi-Layer Perceptron Network for Nonlinear Signal Processing [J]. Journal of Signal Processing Systems, 2002, 32(1): 29-43.

[8.13] Kim T, Adali T, lay. Approximation by Fully Complex Multilayer Perceptrons[J]. Neural Computation, 2003, 15(7): 1641.

[8.14] Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional Neural Networks[C]. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012: 1097-1105.

[8.15] Bengio Y, Lecun Y. Convolutional Networks for Images, Speech, and Time-Series[J]. 1995.

[8.16] Lecun Y, Bengio Y, Hinton G. Deep Learning[J]. Nature, 2015, 521(7553): 436-444.

* http://www.guigu.org/news/hardware/2016052781999.html

[8.17] Lécun Y, Bottou L, Bengio Y, et al. Gradient-based Learning Applied to Document Recognition [J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[8.18] Lehmussola A, Ruusuvuori P, Selinummi J, et al. Computational Framework for Simulating Fluorescence Microscope Images with Cell Populations [J]. IEEE Transactions on Medical Imaging, 2007, 26(7): 1010-1016.

[8.19] Leung H, Haykin S. The Complex Backpropagation Algorithm[J]. IEEE Transactions on Signal Processing, 1991, 39(9): 2101-2104.

[8.20] Mallat S G. A Wavelet Tour of Signal Processing: The Sparse Way[J]. 2009, 31(3): 83-85.

[8.21] Rao A R, Cecchi G A, Peck C C, et al. Unsupervised Segmentation with Dynamical Units[J]. IEEE Transactions on Neural Networks, 2008, 19(1): 168.

[8.22] Reichert D P, Serre T. Neuronal Synchrony in Complex-Valued Deep Networks[J]. Computer Science, 2013.

[8.23] Sainath T N, Kingsbury B, Sindhwani V, et al. Low-rank Matrix Factorization for Deep Neural Network Training with High-dimensional Output Targets[C]. IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2013: 6655-6659.

[8.24] Schmidhuber J. Deep Learning in Neural Networks: An Overview. [J]. Neural Networks, 2015, 61: 85-117.

[8.25] Shalev-Shwartz S, Ben-David S. Understanding Machine Learning: from Theory To algorithms [J]. Lirias. kuleuven. be, 2013.

[8.26] Sutskever I, Martens J, Dahl G, et al. On the Importance of Initialization and Momentum in Deep Learning[C]. International Conference on Machine Learning. 2013.

[8.27] Taigman Y, Yang M, Ranzato M, et al. DeepFace: Closing the Gap to Human-Level Performance in Face Verification[C]. IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2014: 1701-1708.

[8.28] Wager S, Wang S, Liang P. Dropout Training as Adaptive Regularization[J]. Advances in Neural Information Processing Systems, 2013: 351-359.

[8.29] Waibel A, Hanazawa T, Hinton G, et al. Phoneme Recognition Using Time-delay Neural Networks[J]. Readings in Speech Recognition, 1990, 37(3): 393-404.

[8.30] Wan L, Zeiler M, Zhang S, et al. Regularization of Neural Networks Using Dropconnect[C]. International Conference on Machine Learning. 2013: 1058-1066.

[8.31] Courbariaux M, Bengio Y, David J P. Binary Connect: Training Deep Neural Networks with binary weights during propagations[J]. 2016: 3123-3131.

[8.32] Han S, Mao H, Dally W J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding[J]. Fiber, 2016, 56(4): 3-7.

[8.33] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[J]. 2015: 770-778.

[8.34] Hubara I, Soudry D, Ran E Y. Binarized Neural Networks[J]. 2016.

[8.35] Hwang K, Sung W. Fixed-point Feedforward Deep Neural Network Design Using Weights +1, 0, and −1[J]. 2014: 1-6.

[8.36] Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and <0. 5MB Model Size[J]. 2016.

[8.37] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift[J]. Computer Science, 2015.

[8.38] Jia, Yangqing, Shelhamer, et al. Caffe: Convolutional Architecture for Fast Feature Embedding [J]. Eprint Arxiv, 2014: 675-678.

[8.39] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012: 1097-1105.

[8.40] Lécun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[8.41] Lee C Y, Xie S, Gallagher P, et al. Deeply-Supervised Nets[J]. e-Print arXiv, 2014: 562-570.

[8.42] Lin Z, Courbariaux M, Memisevic R, et al. Neural Networks with Few Multiplications[J]. 2016.

[8.43] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector[J]. 2015.

[8.44] Rastegari M, Ordonez V, Redmon J, et al. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks[J]. 2016.

[8.45] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2015: 1-1.

[8.46] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition [J]. Computer Science, 2014.

[8.47] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[J]. 2015: 1-9.

[8.48] Hirose A. Fractal variation of attractors in complex-valued neural networks[J]. Neural Processing Letters, 1994, 1(1): 6-8.

[8.49] Kondo K, Iguchi M, Ishigaki H, et al. Design of complex-valued CNN filters for medical image enhancement[C]//Ifsa World Congress and, Nafips International Conference, 2001. Joint. IEEE Xplore, 2001,3: 1642-1646.

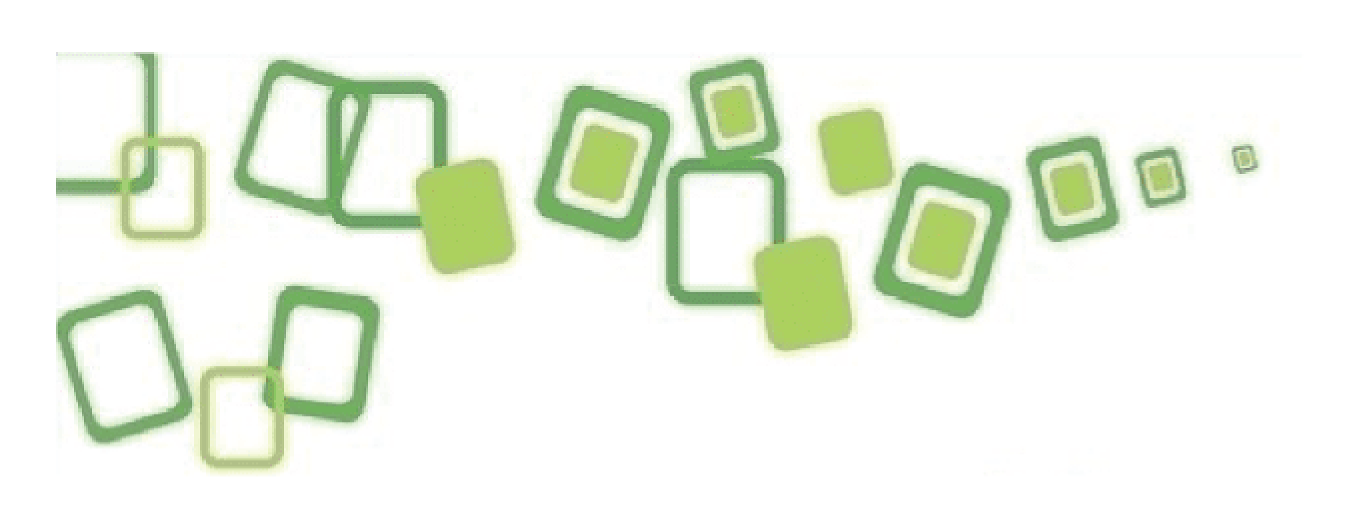
[8.50] Guberman N. On Complex Valued Convolutional Neural Networks[J]. 2016.

[8.51] Tóth G, Földesy P, Roska T. Distance Preserving 1D Turing-Wave Models Via Cnn, Implementation Of Complex-Valued Cnn And Solving A Simple Inverse Pattern Problem (Detection)[J]. Cellular Neural Networks and their Applications, 1996. CNNA-96. Proceedings. 1996 Fourth IEEE Inter, 1996: 109-114.

[8.52] Li F, Zhang B, Liu B. Ternary Weight Networks[J]. 2016.

[8.53] Courbariaux M, Hubara I, Soudry D, et al. Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or −1[J]. 2016.

第9章

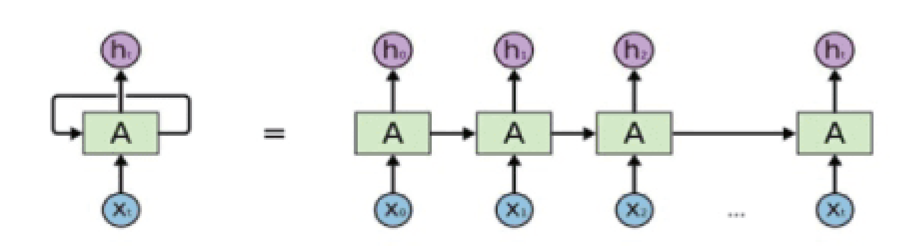


深度循环和递归神经网络

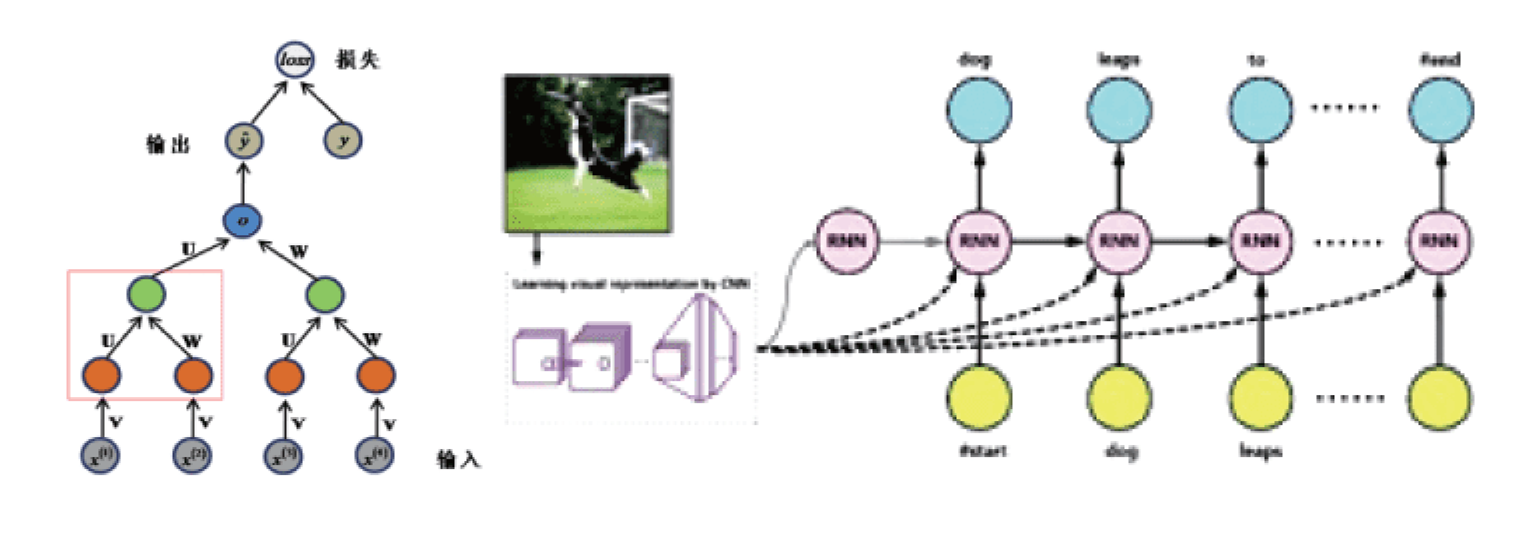
CHAPTER 9

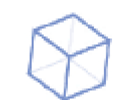
深度神经网络——引入定向循环

■ 深度循环神经网络——时间循环，自然语言处理



■ 深度递归神经网络——结构循环





9.1　深度循环神经网络

传统的深度前馈神经网络在分类、目标识别等任务上取得了出色的表现，但从大脑的生物学功能角度来看，其“仿生”所建立的计算模型应用范围仍有限，如对于分析输入序列（彼此间存在着时间关联性）之间的整体逻辑特性，前面所介绍的深度（前馈）神经网络是无能为力的。本节将介绍的循环神经网络，不同于之前的深度神经网络模型，通过引入（某隐层）定向循环，它能够更好地表征高维度信息的整体逻辑特性。

备注：循环神经网络 Recurrent Neural Networks，递归神经网络 Recursive Neural Networks，通常将递归神经网络简写为 RNN。

9.1.1　循环神经网络的生物机理

在前馈神经网络中，其网络拓扑结构是有向的、无环结构，即连接存在于层与层之间，每层的节点之间是不连接的且前向计算时，层级较高（靠近输出）的隐层不会向层级较低（靠近输入）的隐层定向传播。众所周知，大脑包含数亿万个神经元，而这些神经元又通过百万亿个突触进行连接，尽管揭开这些连接方式看似是不可能完成的任务，但 2015 年底，来自贝勒医学院的研究人员就成功完成了这项任务并将其成果发表在 Science 杂志上，其成果的核心是成功解析了小鼠大脑皮层中神经元的连接方式，并发现大脑皮层中局部回路的基本连线可以通过一系列的互连规则所捕获，且这些规则在大脑皮层中处于不断循环中，从而为理解局部大脑皮层的回路连接提供了一定思路，进一步可以帮助理解大脑的工作原理。

循环神经网络通过使用带有自反馈的神经元，能够处理任意长度的（存在时间关联性）序列；相比传统的深度前馈神经网络，它更加符合生物神经元的连接方式。并且，循环神经网络已经被广泛地应用在自然语言处理等领域，取得了诸多出色的成果。

9.1.2　简单的循环神经网络

下面给出一个简单的循环神经网络（即 Vanilla 循环神经网络）的数学物理描述，网络结构如图 9.1 所示。

从图 9.1 可以看出，循环神经网络类似一个动态系统（即系统的状态按照一定的规律随时间变化的系统）。

1. 数据

$$\{\boldsymbol{x}_t \in \mathbb{R}^n, \boldsymbol{y}_t \in \mathbb{R}^m\}_{t=1}^{T} \tag{9.1}$$

其中 $\boldsymbol{x}_t$ 表示 t 时刻的输入，该时间序列的长度为 T；输出 y_t 与 t 时刻之前（包括 t 时刻）的输入有关系，即

$$\{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_t\} \xrightarrow{\text{关系}} \boldsymbol{y}_t \tag{9.2}$$

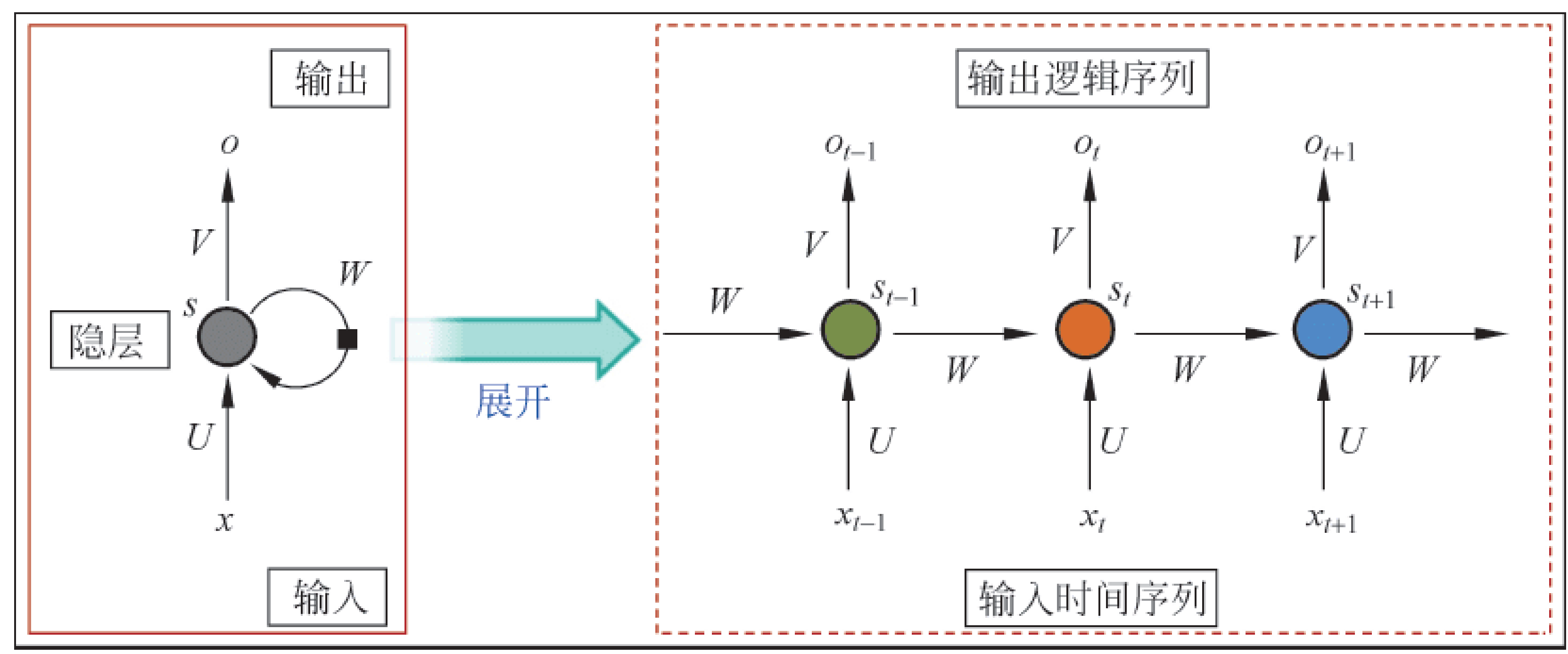


图 9.1　Vanilla 循环神经网络

2. 模型

$$\begin{cases} \boldsymbol{s}_t = \sigma(\boldsymbol{U} \cdot \boldsymbol{x}_t + \boldsymbol{W} \cdot \boldsymbol{s}_{t-1} + \boldsymbol{b}) \\ \boldsymbol{o}_t = \boldsymbol{V} \cdot \boldsymbol{s}_t + \boldsymbol{c} \in \mathbb{R}^m \\ \boldsymbol{y}_t = \text{softmax}(\boldsymbol{o}_t) \in \mathbb{R}^m \end{cases} \tag{9.3}$$

注意这里的 Softmax 不是分类器，而是作为激活函数，即将一个 m 维向量压缩成另一个 m 维的实数向量，其中向量中的每个元素取值都介于(0,1)之间。即

$$\begin{cases} \text{softmax}(\boldsymbol{o}_t) = \dfrac{1}{Z} \cdot [\mathrm{e}^{(o_t(1))}, \cdots, \mathrm{e}^{(o_t(m))}]^{\mathrm{T}} \\ Z = \sum_{j=1}^{m} \mathrm{e}^{(o_t(j))} \end{cases} \tag{9.4}$$

其中 Z 为归一化因子。另外式(9.3)中待优化的参数包括权值连接 $\boldsymbol{U}$、$\boldsymbol{W}$、$\boldsymbol{V}$，及偏置 $\boldsymbol{b}$，$\boldsymbol{c}$，而 $\sigma(\cdot)$为隐层上的激活函数。

3. 优化目标函数

基于关系(9.2)和模型(9.3)，利用负对数似然(即交互熵)建构损失函数，继而得到的优化目标函数为：

$$\begin{aligned} \min_{\theta} J(\theta) &= \sum_{t=1}^{T} \text{loss}(\hat{\boldsymbol{y}}_t, \boldsymbol{y}_t) \\ &= \sum_{t=1}^{T} \left(- \left[\sum_{j=1}^{m} y_t(j) \cdot \log(\hat{y}_t(j)) + (1 - y_t(j)) \cdot \log(1 - \hat{y}_t(j)) \right] \right) \end{aligned} \tag{9.5}$$

其中 $y_t(j)$为 $\boldsymbol{y}_t$ 的第 j 个元素，参数 θ 为：

$$\theta = [\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}; \boldsymbol{b}, \boldsymbol{c}] \tag{9.6}$$

4. 求解

由于循环神经网络在每一个 $t(t=1,2,\cdots,T)$时刻对应着一个监督信息 y_t，相应的损失



项简记为：

$$J_t(\theta)=\text{loss}(\hat{\boldsymbol{y}}_t,\boldsymbol{y}_t) \tag{9.7}$$

关于优化目标函数，即式(9.5)的求解，可以通过时间反向传播算法实现。其核心是如下五个偏导数的求解：

$$\left[\frac{\partial J(\theta)}{\partial \boldsymbol{V}},\frac{\partial J(\theta)}{\partial \boldsymbol{c}},\frac{\partial J(\theta)}{\partial \boldsymbol{W}},\frac{\partial J(\theta)}{\partial \boldsymbol{U}},\frac{\partial J(\theta)}{\partial \boldsymbol{b}}\right] \tag{9.8}$$

其中前两项偏导数的求解，依据如下误差传播项的求解：

$$\boldsymbol{\delta}_{\boldsymbol{o}_t}=\frac{\partial J_t(\theta)}{\partial \boldsymbol{o}_t} \tag{9.9}$$

注意$\boldsymbol{\delta}_{\boldsymbol{o}_t}$是$t$时刻的目标函数(式(9.7))关于$t$时刻的输出$\boldsymbol{o}_t$的偏导数。后面三项偏导数的求解，则根据误差传播项：

$$\boldsymbol{\delta}_{\boldsymbol{s}_t}=\frac{\partial J_t(\theta)}{\partial \boldsymbol{s}_t} \tag{9.10}$$

解释同上；由于篇幅所限，这里仅给出目标函数关于$\boldsymbol{W}$的偏导数，即

$$\frac{\partial J(\theta)}{\partial \boldsymbol{W}}=\sum_{t=1}^{T}\sum_{k=1}^{t}\frac{\partial \boldsymbol{s}_k}{\partial \boldsymbol{W}}\cdot\frac{\partial \boldsymbol{s}_t}{\partial \boldsymbol{s}_k}\cdot\frac{\partial J_t(\theta)}{\partial \boldsymbol{s}_t}=\sum_{t=1}^{T}\sum_{k=1}^{t}\frac{\partial \boldsymbol{s}_k}{\partial \boldsymbol{W}}\cdot\frac{\partial \boldsymbol{s}_t}{\partial \boldsymbol{s}_k}\cdot\delta_{s_t} \tag{9.11}$$

注意，由于隐层的t时刻输出s_t与之前的输出$\boldsymbol{s}_k(k=1,2,\cdots,t)$有关系，即见模型中式(9.3)，而参数$\boldsymbol{W}$恰好是这种关系的内蕴。其中依据链式法则有：

$$\frac{\partial \boldsymbol{s}_t}{\partial \boldsymbol{s}_k}=\prod_{j=k+1}^{t}\frac{\partial \boldsymbol{s}_j}{\partial \boldsymbol{s}_{j-1}}=\prod_{j=k+1}^{t}(\boldsymbol{W}^{\mathrm{T}}\cdot\text{diag}(\sigma'(\boldsymbol{s}_{j-1}))) \tag{9.12}$$

其中的$\sigma'(\cdot)$为激活函数$\sigma(\cdot)$的导数，另外，函数diag(·)为向量扩展矩阵，即形成的矩阵以向量为对角元素。

实验中，式(9.3)中隐层输出中的激活函数$\sigma(\cdot)$常取Tanh(·)函数，在训练后期，式(9.10)和式(9.12)所对应的梯度变的比较小，进一步，连乘后的梯度值使得式(9.11)变的更小，容易出现梯度弥散现象；若$\sigma(\cdot)$取Sigmoid(·)函数也会发生同样的情形。常用避免梯度弥散的技巧包括：参数初始化策略，以及使用ReLU函数作为激活函数等。

注意：虽然循环神经网络从理论上可以建立长时间间隔的状态之间的依赖关系，但由于梯度弥散或梯度爆炸(即连乘后的梯度值趋于无穷大，造成系统不稳定)问题，实际应用中，只能学习到短周期(或使用马尔科夫链)的依赖关系，这便是所谓的长期依赖问题。后面将针对此问题，描述长短时记忆网络。

9.1.3　深度循环神经网络的数学物理描述

深度循环神经网络中的“深度”，与传统的深度神经网络不同，它指时间和空间(如网络中的隐层个数)特性上的深度，其设计模式包括以下三种：

- 每个“时刻”都有输出，并且隐层引入定向循环，如简单循环神经网络，见图9.1。
- 每个“时刻”都有输出，且该时刻的输出到下一时刻的隐层之间有循环连接，见图9.2。
- 隐层之间存在着循环连接，但输出仅出现在若干个时刻后，不再每一时刻都对应着输出，见图9.3。

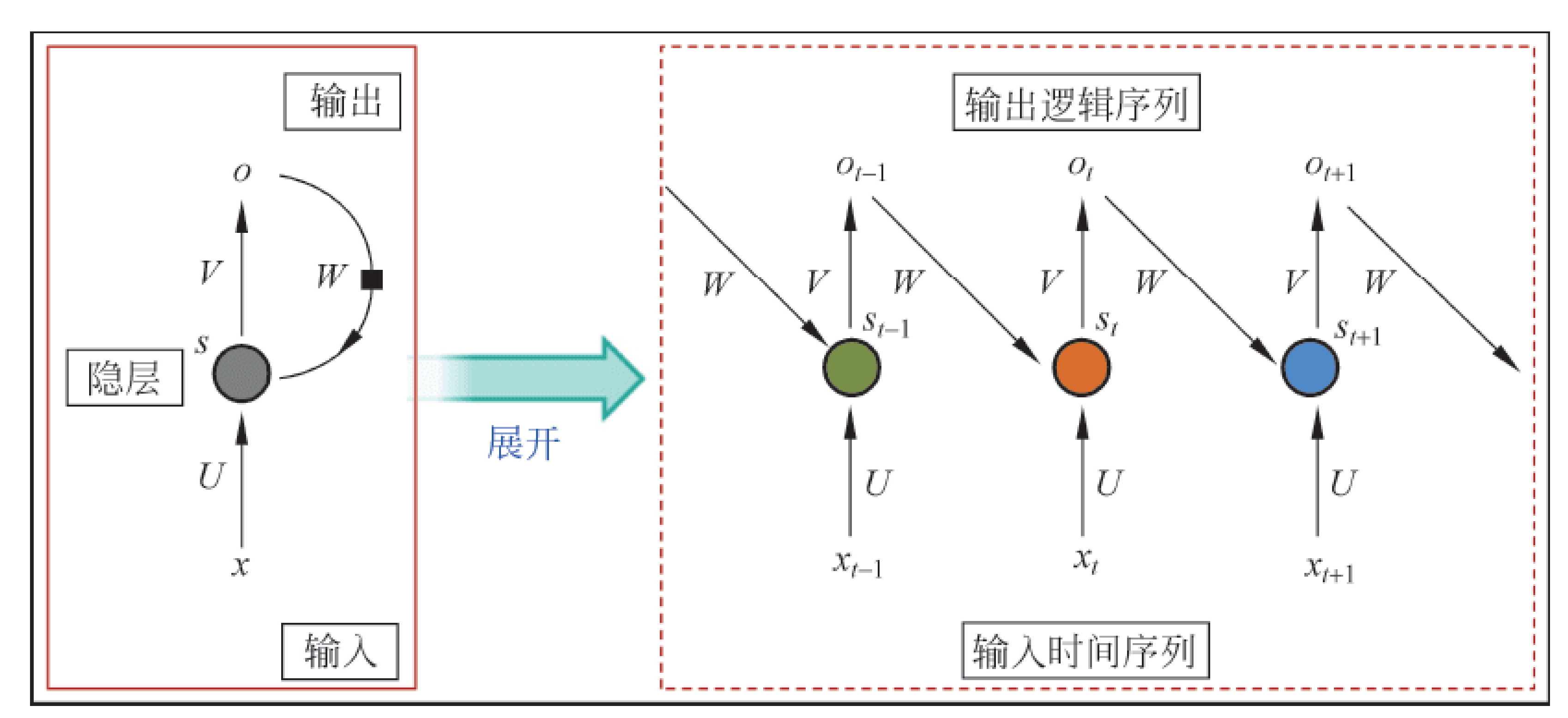


图 9.2　深度循环神经网络——输出与隐层定向循环

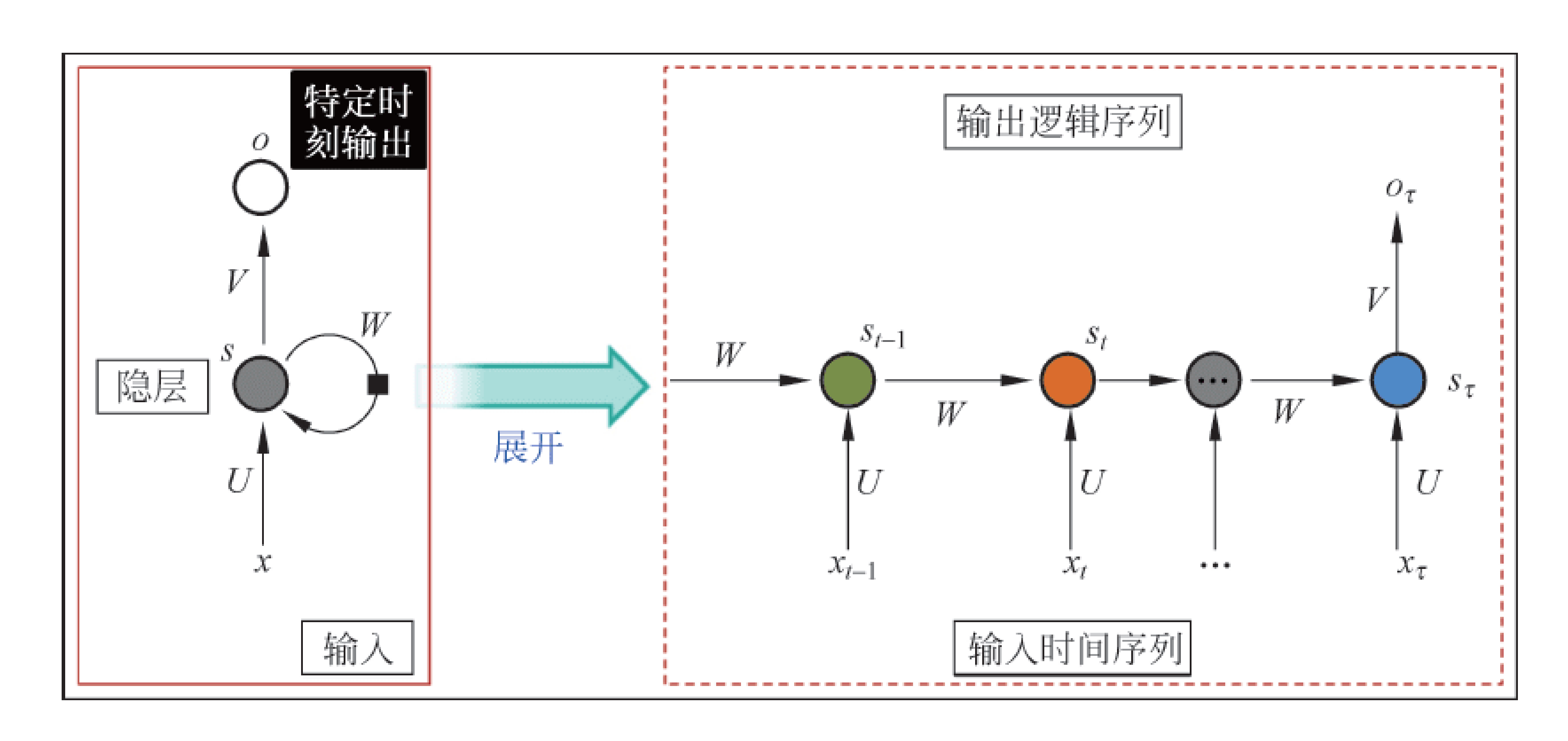


图 9.3　深度循环神经网络——隐层间定向循环且定向时刻输出

值得注意的是：传统的深度神经网络中卷积神经网络的特点是局部连接、权值共享和局部平移不变特性，其中的权值共享意味着共享计算；而循环神经网络中随着“时间”深度的加深对参数实行“平流移植”来实现共享计算。另外，序列中时间特性的依赖关系通常会引起长期依赖问题(即记忆能力受限)。

前面的简单循环神经网络给出了单隐层结构下关于时间特性的模型设计；接下来介绍带有空间特性的循环神经网络，即通过增加隐层个数，来提升网络的表达能力，见图 9.4，给出常见的三种通过增加隐层和定向循环的深度循环神经网络。实验发现将循环神经网络的隐层(状态)设为多层的好处在于表征能力的提升，但需注意的是增加深度会导致优化困难，从而使得模型的实际效果变差。

另外，实际应用中，根据输入与输出之间的关系，常见的网络模型大致可分为如下 5 类，即图 9.5 所示，注意这里的输入与输出为向量或矩阵。

- 一对一的方式：本质上并没有使用循环神经网络(没有定向循环)，其应用场景大多为图像分类。

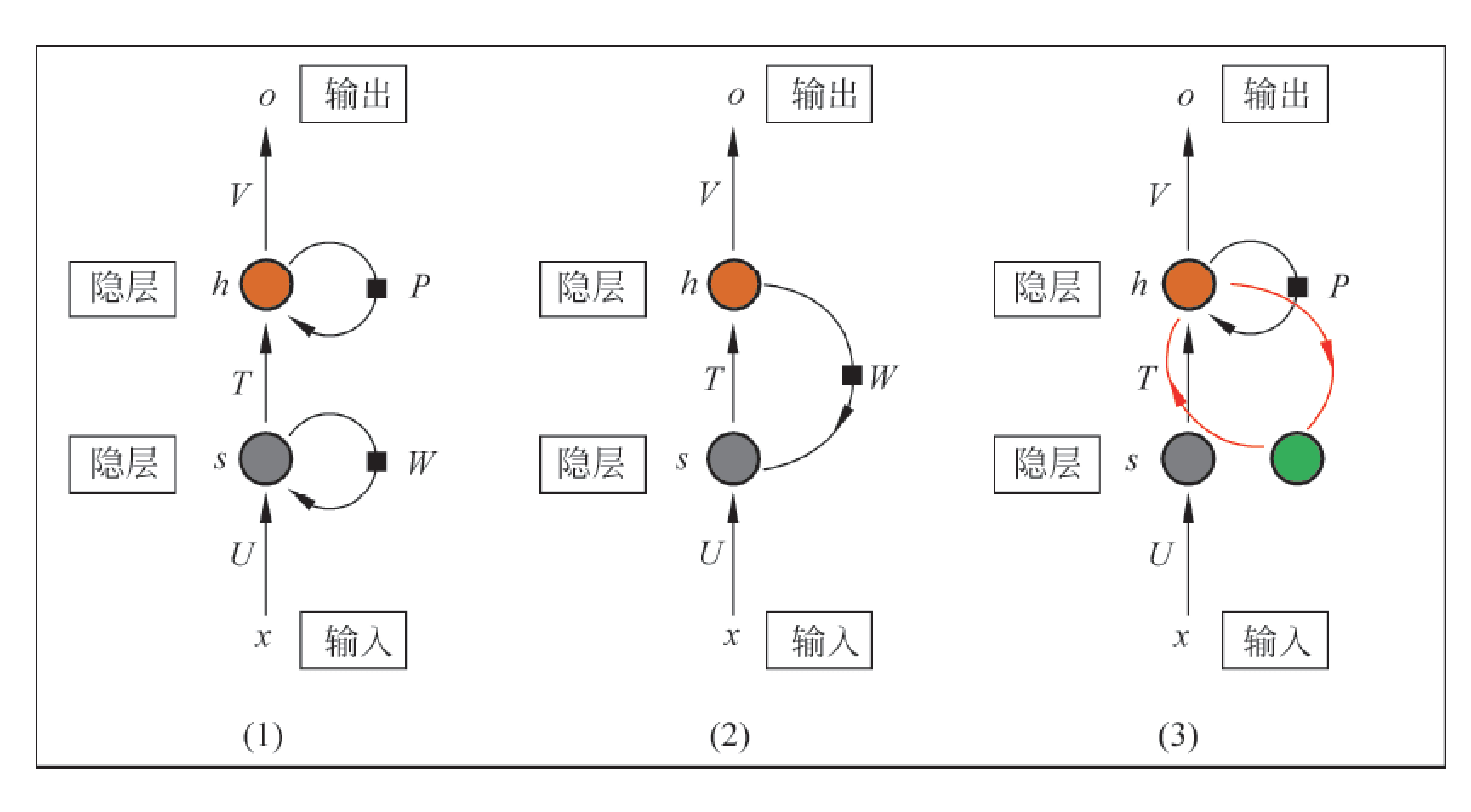


图 9.4　深度循环神经网络——增加隐层与定向循环

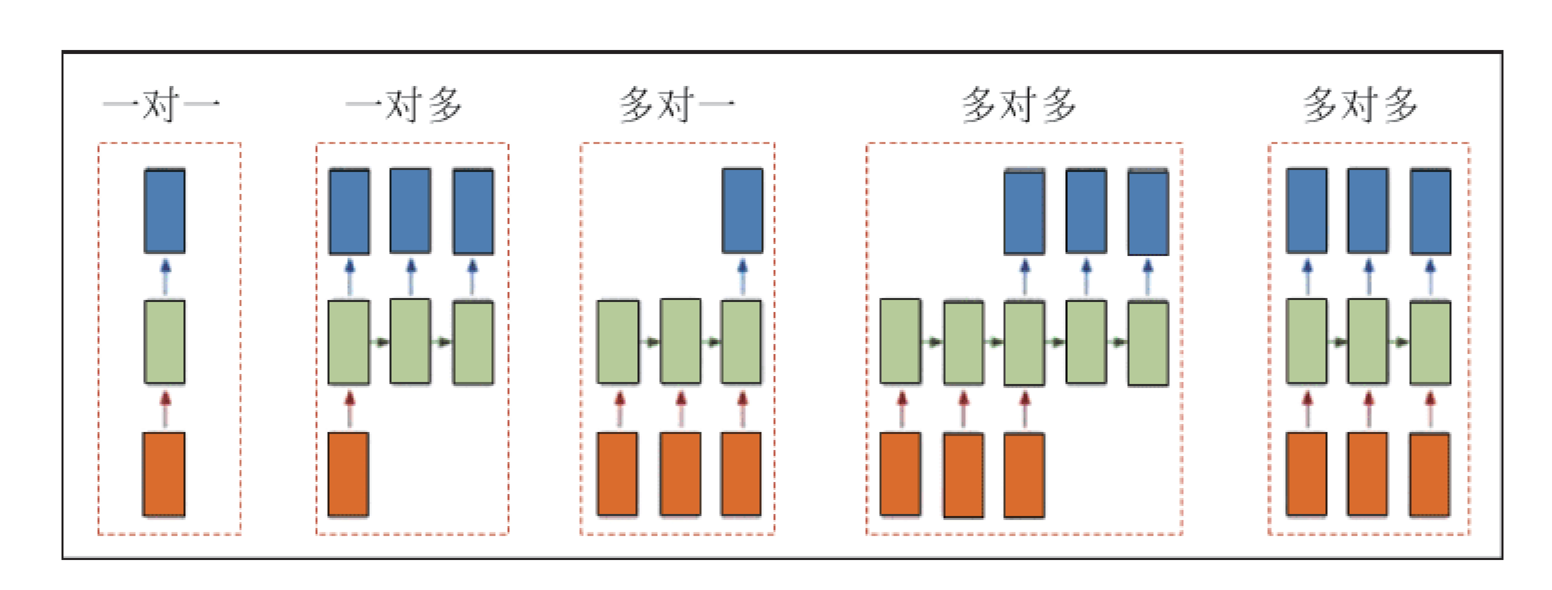


图 9.5　深度循环神经网络——输入与输出之间的关系

■ 一对多的方式：可用于图片字幕的生成，即输入为图片，输出为一段文本或文字序列。
■ 多对一的方式：常用于文本生成图片或情感分析(即给定一段文本，将其分为积极或消极的情感)。
■ 左边的多对多的方式：常用于机器翻译(即将中文语句翻译为英文)。
■ 右边的多对多的方式：常用于视频分类，即为每一帧打上标签或标注。

备注：深度循环神经网络常被用于自然语言处理，以期挖掘序列间的逻辑特性及潜在的对应关系. 基于以上的关系，可以架构具有多隐层的深度循环神经网络。

为了避免深度循环神经网络过早陷入局部最优，参数初始化常被作为一种有效的策略以用来抑制梯度弥散现象的发生。问题是能否将深度堆栈网络中基于自编码器的逐层学习方式移植到图 9.4 中这些深度循环神经网络中？不同的应用任务对应着不同的初始化策略。下面基于受限玻尔兹曼机的循环神经网络，简要地给出一种初始化策略，如图 9.6 所示。

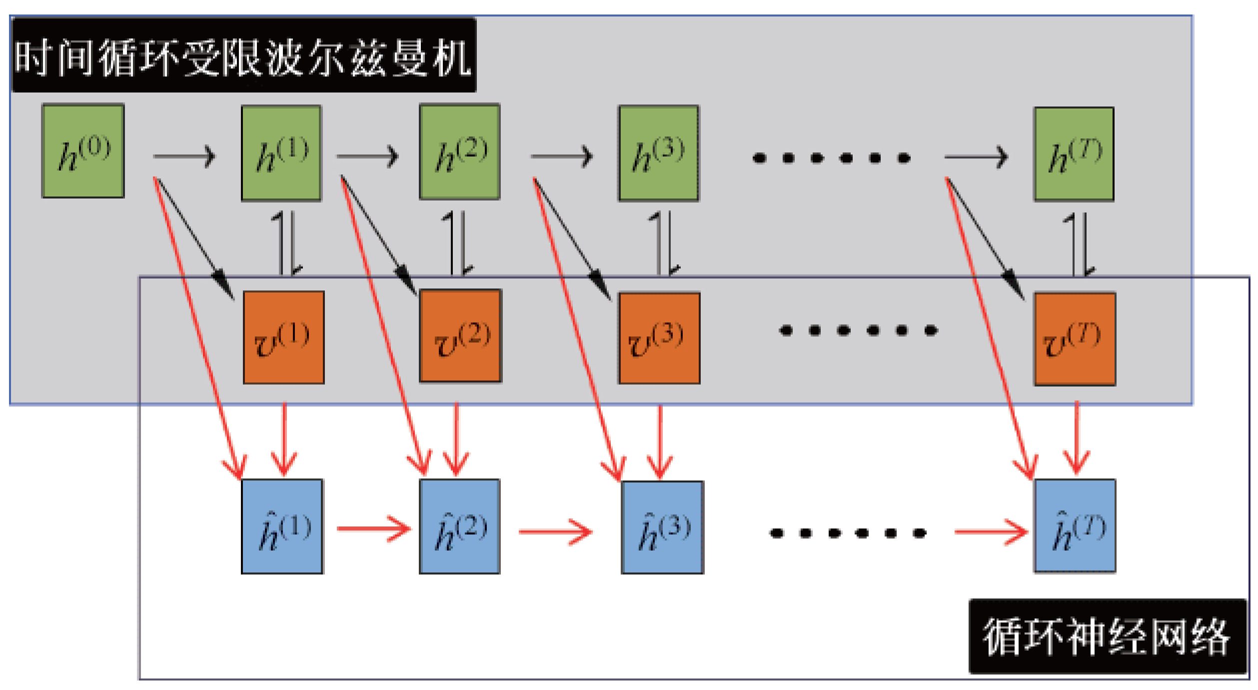


图 9.6　基于受限玻尔兹曼机的循环神经网络 RNN-RBM

1. 数据(训练数据集)

$$\{\boldsymbol{v}^{(t)} \in \mathbb{R}^m\}_{t=1}^{T} \tag{9.13}$$

2. 模型

网络模型包括两部分,一部分是时间循环受限玻尔兹曼机,即 Recurrent Temporal Restrict Boltzmann Machine,简记为 RTRBM;另一部分为循环神经网络。

1) RTRBM(注意隐层引入定向循环结构)

首先,给出 $t-1$ 时刻的隐层输出 $\boldsymbol{h}^{(t-1)}$ 到下一时刻的隐层和可视层的状态转移关系,即

$$\begin{cases} \boldsymbol{b}_h^{(t)} = \boldsymbol{W}_h \cdot \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_h \\ \boldsymbol{b}_v^{(t)} = \boldsymbol{W}_v \cdot \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_v \end{cases} \tag{9.14}$$

其中 $\boldsymbol{b}_h^{(t)}$ 为 t 时刻隐层的状态,$\boldsymbol{b}_v^{(t)}$ 为 t 时刻可视层的状态,另外待优化的参数包括权值参数 $(\boldsymbol{W}_h, \boldsymbol{W}_v)$ 和偏置 $(\boldsymbol{b}_h, \boldsymbol{b}_v)$,进一步 t 时刻的隐层输出为(由于隐层引入定向循环结构):

$$\boldsymbol{h}^{(t)} = \sigma(\boldsymbol{W} \cdot \boldsymbol{v}^{(t)} + \boldsymbol{b}_h^{(t)}) = \sigma(\boldsymbol{W} \cdot \boldsymbol{v}^{(t)} + \boldsymbol{W}_h \cdot \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_h) \tag{9.15}$$

其中的 $\boldsymbol{W}$ 为受限玻尔兹曼机所求出的权值连接矩阵。

而 t 时刻可视层的输出为:

$$\hat{\boldsymbol{v}}^{(t)} = \sigma(\boldsymbol{b}_v^{(t)}) = \sigma(\boldsymbol{W}_v \cdot \boldsymbol{h}^{(t-1)} + \boldsymbol{b}_v) \tag{9.16}$$

可知,在整个 RTRBM 部分,待优化的参数为:

$$\theta = [\boldsymbol{W}, \boldsymbol{W}_h, \boldsymbol{W}_v, \boldsymbol{b}_h, \boldsymbol{b}_v] \tag{9.17}$$

其中当 $t=1$,隐层的输出 $\boldsymbol{h}^{(t-1)}$ 退化为 $\boldsymbol{h}^{(0)}$。

2) 循环神经网络

$$\hat{\boldsymbol{h}}^{(t)} = \sigma(\boldsymbol{U} \cdot \boldsymbol{v}^{(t)} + \boldsymbol{R} \cdot \hat{\boldsymbol{h}}^{(t-1)} + \boldsymbol{b}_{\hat{h}}) \tag{9.18}$$

注意该部分 $t-1$ 时刻的隐层输出 $\hat{\boldsymbol{h}}^{(t-1)}$ 可利用 RTRBM 部分的 $\boldsymbol{h}^{(t-1)}$ 来替代。可知待优化的参数为:

$$\vartheta = [\boldsymbol{U}, \boldsymbol{R}, \boldsymbol{b}_{\hat{h}}] \tag{9.19}$$

3. 优化目标函数

关于 RTRBM 部分的优化目标函数，根据式(9.16)估计的输出和式(9.13)对应的真实输入，利用交互熵可得：

$$\min_{\theta} J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{n_v}[v_j^{(t)}\log(\hat{v}_j^{(t)}) + (1-v_j^{(t)})\log(1-\hat{v}_j^{(t)})] \tag{9.20}$$

注意 $v_j^{(t)}$ 为 t 时刻 $\boldsymbol{v}^{(t)} \in \mathbb{R}^m$ 的第 j 个元素，其中 $j=1,2,\cdots,m$；待优化的参数为式(9.17)，这一步仅作为参数 θ 的初始化。

关于循环神经网络部分，利用 RTRBM 部分所获取的数据，即式(9.15)和式(9.16)，得到(逼近)数据集描述，即

$$\{\hat{\boldsymbol{v}}^{(t)}, \boldsymbol{h}^{(t)}\} \tag{9.21}$$

再利用循环神经网络模型，即式(9.18)，将其中的 $\boldsymbol{v}^{(t)}$ 利用 $\hat{\boldsymbol{v}}^{(t)}$ 来替代，类似之前简单的循环神经网络的分析，需要注意的是这里的 $\boldsymbol{h}^{(t)}$ 为 RTRBM 部分由公式(9.15)得到的，而 $\hat{\boldsymbol{h}}^{(t)}$ 为循环神经网络部分由式(9.18)估计得到的，所以建立如下的优化目标函数：

$$\min_{\vartheta} J(\vartheta) = \sum_{t=1}^{T}\text{loss}(\hat{\boldsymbol{h}}^{(t)}, \boldsymbol{h}^{(t)}) \tag{9.22}$$

这一步仅作为参数 ϑ 的初始化。

最后，根据 RBM 可知，该网络的中心为可视化层，其他层与可视层之间的关系可由模型部分获悉。综上，整个 RNN-RBM 网络的优化目标函数为：

$$\min_{(\theta,\vartheta)} J(\theta,\vartheta) = -\frac{1}{T}\sum_{t=1}^{T}\log(P(\boldsymbol{v}^{(t)})) \tag{9.23}$$

其中 $P(\boldsymbol{v}^{(t)})$是以 $\boldsymbol{v}^{(t)}$ 为输出的分布估计(其与受限玻尔兹曼机中的目标函数构造一致)，由于输入是时间相关的序列，所以它待优化的参数为(θ,ϑ)。这一部分可视为 RTRBM 和循环神经网络这两部分的博弈。

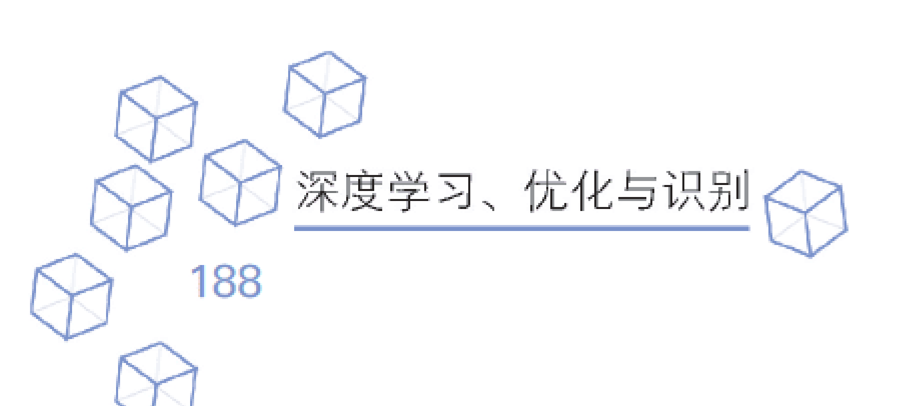
4. 求解

整体优化目标函数(9.23)关于参数(θ,ϑ)的偏导数求解可参考第 1 章，而这里仅给出简要的框架(分为三步)，并强调参数初始化的策略。

(1) 根据循环神经网络参数 ϑ 初始化，利用式(9.18)求解隐层的输出$\hat{\boldsymbol{h}}^{(t)}$。

(2) 在 RTRBM 部分，利用 $\boldsymbol{h}^{(t)}$(由$\hat{\boldsymbol{h}}^{(t)}$逼近替代)和输入 $\boldsymbol{v}^{(t)}$，并利用式(9.16)得到输入的估计 $\hat{\boldsymbol{v}}^{(t)}$，再利用式(9.20)更新参数 θ。

(3) 根据 RTRBM 部分所获取的数据，即式(9.21)，基于优化目标函数来更新循环神经网络参数 ϑ。依此三步交替进行，直至整个网络状态达到平衡。



9.2 深度递归神经网络

可以看出，与链式结构的循环神经网络不同，递归神经网络是通过带有树状相似的神经网络结构来递归构造复杂的深度网络的。本质上，递归神经网络是对循环神经网络的一个有效扩展，但由于二者的构造方式不同，所以具有不同类型的计算图。

9.2.1 简单的递归神经网络

递归神经网络由 Pollack 于 1990 年引入，而 Bottou 于 2011 年描述了这类网络的潜在用途——学习序列中的逻辑推理。下面给出一个简单的递归神经网络结构，见图 9.7。

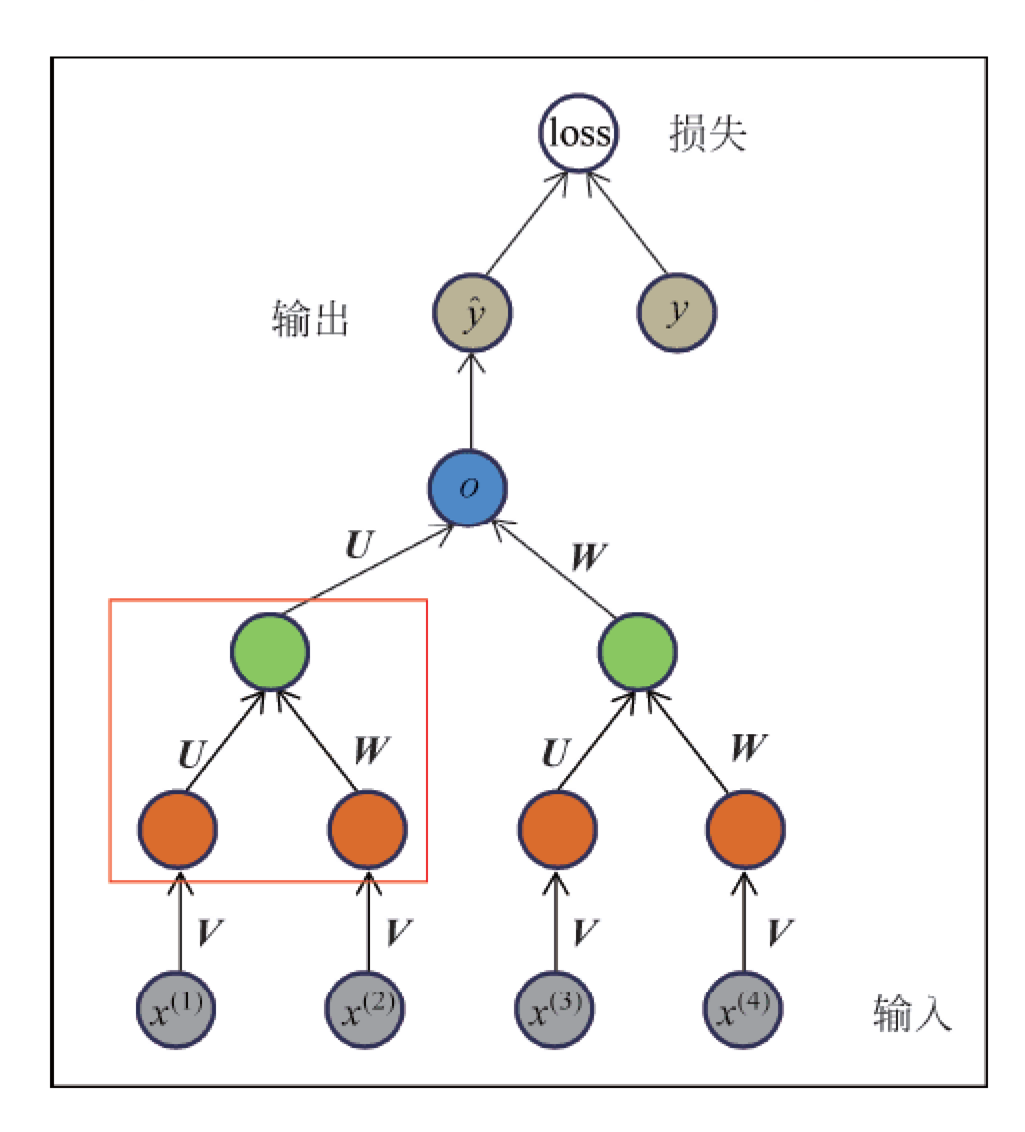


图 9.7 简单的递归神经网络

1. 数据

训练数据集为：

$$\{\boldsymbol{x}_t, \boldsymbol{y}_t\}_{t=1}^{T} \tag{9.24}$$

其中 t 时刻的输入有：

$$\boldsymbol{x}_t = [\boldsymbol{x}_t^{(1)}, \boldsymbol{x}_t^{(2)}, \boldsymbol{x}_t^{(3)}, \boldsymbol{x}_t^{(4)}] \tag{9.25}$$

需要注意的是 $\boldsymbol{x}_t^{(1)}$ 并不一定是标量，且该输入序列的长度为 4。

2. 模型

依据图 9.7，可以给出输入与输出之间的关系为：



$$\begin{cases} \boldsymbol{h} = [\boldsymbol{h}^{(1)}, \boldsymbol{h}^{(2)}, \boldsymbol{h}^{(3)}, \boldsymbol{h}^{(4)}] \\ \boldsymbol{s} = [\boldsymbol{s}^{(1)}, \boldsymbol{s}^{(2)}] \\ \boldsymbol{y} = \sigma(\boldsymbol{o}) \end{cases} \tag{9.26}$$

其中第一隐层 $\boldsymbol{h}$ 中每一个“元”有：

$$\boldsymbol{h}^{(i)} = \sigma(\boldsymbol{V} \cdot \boldsymbol{x}^{(i)} + \boldsymbol{b}) \tag{9.27}$$

且 $i=1,2,3,4$；第二隐层 $\boldsymbol{s}$ 中每一个“元”有：

$$\boldsymbol{s}^{(j)} = \sigma(\boldsymbol{U} \cdot \boldsymbol{h}^{(2\cdot j-1)} + \boldsymbol{W} \cdot \boldsymbol{h}^{(2\cdot j)} + \boldsymbol{c}) \tag{9.28}$$

且 $j=1,2$；最后输出层的状态有：

$$\boldsymbol{o} = \sigma(\boldsymbol{U} \cdot \boldsymbol{s}^{(1)} + \boldsymbol{W} \cdot \boldsymbol{s}^{(2)} + \boldsymbol{d}) \tag{9.29}$$

注意公式中出现的参数 $\boldsymbol{b},\boldsymbol{c},\boldsymbol{d}$ 均为偏置，$\sigma(\cdot)$为激活函数。

3. 优化目标函数

根据输入与输出之间的关系，对于每一时刻都有对应着的损失，即

$$J_t(\theta) = \text{loss}(\hat{\boldsymbol{y}}^{(t)}, \boldsymbol{y}^{(t)}) \tag{9.30}$$

其中损失函数常用交互熵的形式给出，当然也可以利用能量范数损失构造。待优化的参数为：

$$\theta = (\boldsymbol{V}, \boldsymbol{U}, \boldsymbol{W}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}) \tag{9.31}$$

在数据集上优化目标函数为：

$$\min_{\theta} J(\theta) = \frac{1}{T}\sum_{t=1}^{T} J_t(\theta) + \lambda \cdot R(\theta) \tag{9.32}$$

其中 $R(\theta)$为正则项，即有：

$$R(\theta) = \|\boldsymbol{V}\|_F^2 + \|\boldsymbol{U}\|_F^2 + \|\boldsymbol{W}\|_F^2 \tag{9.33}$$

4. 求解

与循环神经网络一致，递归神经网络也利用随机梯度的方式实现参数的更新与求解，这里不再赘述。

9.2.2　深度递归神经网络的优势

深度递归神经网络的一个明显优势是：对于长度为 τ 的序列，深度(通过非线性操作的组合数量来衡量)可以急剧地从 τ 下降至 $O(\log(\tau))$，这可能有助于解决长期依赖问题。在实际应用中，相比较于循环神经网络，递归神经网络通常被用于基于词嵌入的短语和句子的连续表示，并且在自然场景图像(图 9.8)和自然语言处理中的学习序列和树结构方面(图 9.9)取得成功。



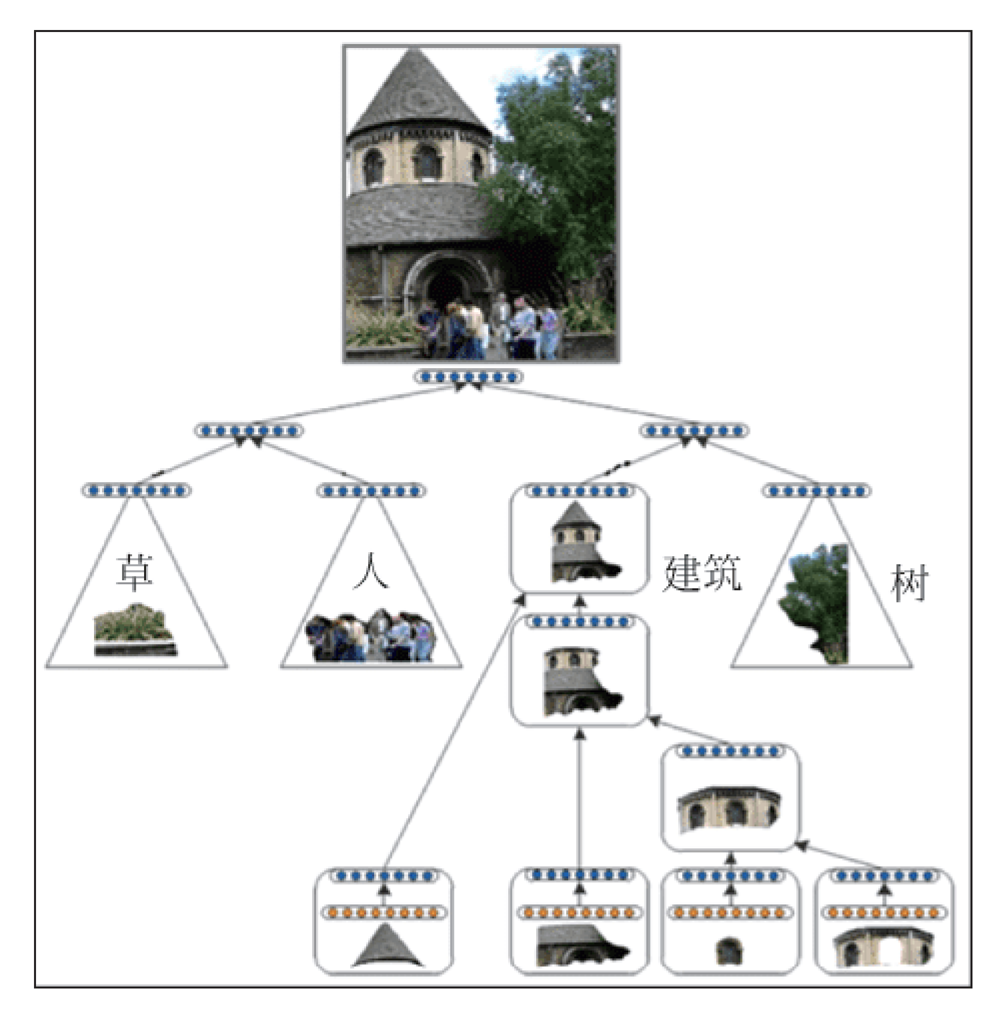


图 9.8　基于深度递归神经网络的自然场景图像剖析

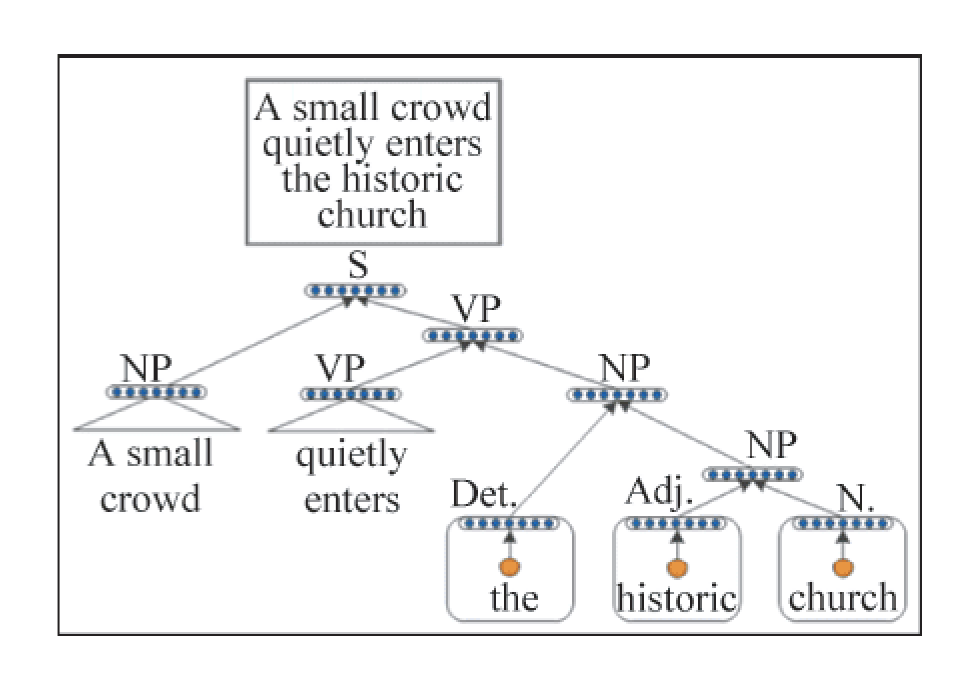


图 9.9　基于深度递归神经网络的自然语言句子分析

9.3　长短时记忆神经网络

已知(简单的)循环神经网络的核心问题是随着时间间隔的增加(即 Long Term Dependencies)容易出现梯度爆炸或梯度弥散,为了有效地解决这一问题通常引入门限机制来控制信息的累积速度,并可以选择遗忘之前的累积信息。而这种门限机制下的循环神经网络包括长短时记忆神经网络和门限循环单元网络。这里将重点给出长短时记忆神经网络的数学分析。

注意:长短时记忆神经网络是循环神经网络的一个变体。

9.3.1　改进动机分析

在简单的循环神经网络中,从式(9.11)和式(9.12)中知,若定义:

$$\zeta = \boldsymbol{W}^{\mathrm{T}} \cdot \mathrm{diag}(\sigma'(\boldsymbol{s}_{j-1})) \tag{9.34}$$

则有:

$$\prod_{j=k+1}^{t} (\boldsymbol{W}^{\mathrm{T}} \cdot \mathrm{diag}(\sigma'(\boldsymbol{s}_{j-1}))) \longrightarrow \zeta^{t-k} \tag{9.35}$$

如果 ζ 的谱半径 $\|\zeta\|>1$,当时差$(t-k)$趋于无穷大时,则式(9.35)会发散并且导致系统出现所谓的梯度爆炸的问题;相反,若 $\|\zeta\|<1$,则会随着时差的无限扩大而导致梯度弥散的问题。

为了避免梯度爆炸或梯度弥散的问题,核心是将 ζ 的谱半径设为 $\|\zeta\|=1$,不失一般

性，若将 $\boldsymbol{W}$ 设为单位矩阵，同时 $\sigma'(\boldsymbol{s}_{j-1})$ 的谱范数也为 1，即模型（即式(9.3)）隐层的关系退化为：

$$\boldsymbol{s}_t = \sigma(\boldsymbol{U} \cdot \boldsymbol{x}_t + \boldsymbol{W} \cdot \boldsymbol{s}_{t-1} + \boldsymbol{b}) \xrightarrow{\text{退化}} \boldsymbol{s}_t = \boldsymbol{U} \cdot \boldsymbol{x}_t + \boldsymbol{s}_{t-1} + \boldsymbol{b} \tag{9.36}$$

但这样的形式，丢失了非线性激活的性质。因此，改进后的方式是引入一个新的状态，记为 c_t，来进行信息的非线性传递，即

$$\begin{cases} c_t = c_{t-1} + \boldsymbol{U} \cdot x_t \\ \boldsymbol{s}_t = \tanh(c_t) \end{cases} \tag{9.37}$$

注意这里的非线性激活函数为 tanh(·)。

注意：随着时间 t 的增加，c_t 的累积量将会变得越来越大（见图 9.10），为了解决这个问题，引入了门限机制，以期控制信息的累积速度，并可以选择遗忘部分之前累积的信息，这便是长短时记忆神经网络。

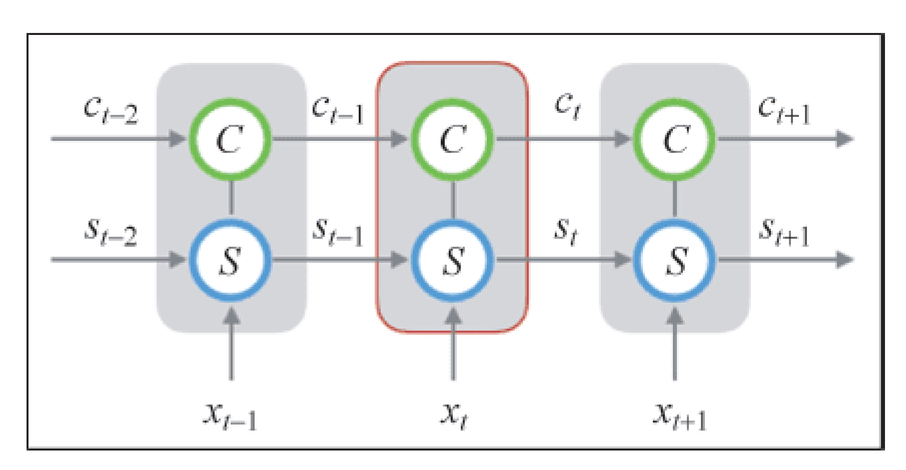


图 9.10　增加新状态后的循环神经网络结构

9.3.2　长短时记忆神经网络的数学分析

基于图 9.10，长短时记忆神经网络的核心是设计这个新状态 C，以期控制信息的变化。注意，图 9.10 中 t 时刻的输入包括三个，即 $\boldsymbol{x}_t$，$\boldsymbol{c}_{t-1}$ 和 $\boldsymbol{s}_{t-1}$；输出包括两个，即 $\boldsymbol{c}_t$ 和 $\boldsymbol{s}_t$；长短时记忆神经网络的结构包括以下两点：

- 关于状态 C，通过遗忘门确定 $\boldsymbol{c}_{t-1}$ 有多少成分保留在 $\boldsymbol{c}_t$ 中，以及通过输入门确定 $\boldsymbol{x}_t$ 中有多少成分保留在 $\boldsymbol{c}_t$ 中。
- 关于状态 S，输出门通过控制单元 $\boldsymbol{c}_t$ 确定输出 $\boldsymbol{o}_t$ 中有多少成分输出到 $\boldsymbol{s}_t$。

注意网络的核心设计包括三个门，即输入门、遗忘门和输出门。具体每一个门输入、门限与输出的数学分析和网络结构如图 9.11 所示。

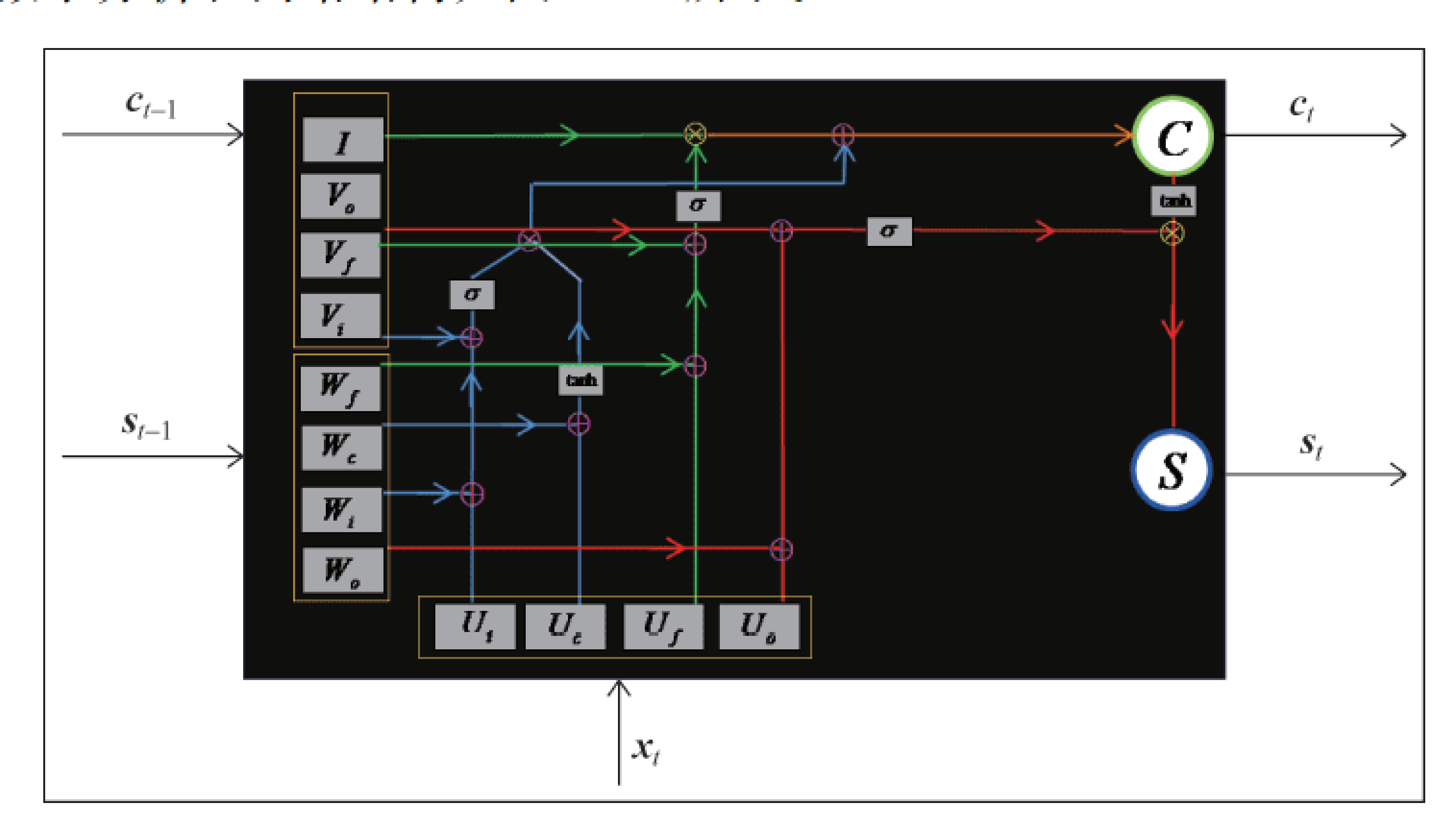


图 9.11　长短时记忆神经网络的标准模块



1. 输入门

该门的主要目的是确定输入 $\boldsymbol{x}_t$ 中有多少成分保留在 $\boldsymbol{c}_t$ 中，实现公式为：

$$\begin{cases} \boldsymbol{i}_t = \sigma(\boldsymbol{U}_i \cdot \boldsymbol{x}_t + \boldsymbol{W}_i \cdot \boldsymbol{s}_{t-1} + \boldsymbol{V}_i \cdot \boldsymbol{c}_{t-1}) \\ \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{U}_c \cdot \boldsymbol{x}_t + \boldsymbol{W}_c \cdot \boldsymbol{s}_{t-1}) \end{cases} \tag{9.38}$$

这里的 i 代表“input”；其中 $\boldsymbol{i}_t$ 为 t 时刻输入门的输入，通过输入门，将输入中对应的$\tilde{\boldsymbol{c}}_t$ 倍保留下来，即输入门过后，保留在 $\boldsymbol{c}_t$ 中的成分为 $\boldsymbol{i}_t \otimes \tilde{\boldsymbol{c}}_t$，其中符号“⊗”表示对应向量中对应元素相乘。

2. 遗忘门

该门的目的是确定 t 时刻输入中的 $\boldsymbol{c}_{t-1}$ 有多少成分保留在 $\boldsymbol{c}_t$ 中，实现公式为：

$$\boldsymbol{f}_t = \sigma(\boldsymbol{U}_f \cdot \boldsymbol{x}_t + \boldsymbol{W}_f \cdot \boldsymbol{s}_{t-1} + \boldsymbol{V}_f \cdot \boldsymbol{c}_{t-1}) \tag{9.39}$$

这里的 f 代表“forget”；这个公式是遗忘门的门限，与输入门的门限$\tilde{\boldsymbol{c}}_t$ 一样，即通过遗忘门之后，保留在 $\boldsymbol{c}_t$ 中的成分为 $\boldsymbol{f}_t \otimes \boldsymbol{c}_{t-1}$。

3. 输出门

该门的目的是利用控制单元 $\boldsymbol{c}_t$ 确定输出 $\boldsymbol{o}_t$ 中有多少成分输出到隐层 $\boldsymbol{s}_t$ 中；首先，经过输入门与遗忘门之后的状态 C，即 $\boldsymbol{c}_t$ 实现公式为：

$$\boldsymbol{c}_t = \boldsymbol{i}_t \otimes \tilde{\boldsymbol{c}}_t + \boldsymbol{f}_t \otimes \boldsymbol{c}_{t-1} \tag{9.40}$$

其中前一项为输入门后保留在 $\boldsymbol{c}_t$ 中的成分，后一项是遗忘门后保留在 $\boldsymbol{c}_t$ 中的成分。其次，为了确定 $\boldsymbol{c}_t$ 有多少成分保留在 $\boldsymbol{s}_t$ 中，先给出输出的实现公式为：

$$\boldsymbol{o}_t = \sigma(\boldsymbol{U}_o \cdot \boldsymbol{x}_t + \boldsymbol{W}_o \cdot \boldsymbol{s}_{t-1} + \boldsymbol{V}_o \cdot \boldsymbol{c}_t) \tag{9.41}$$

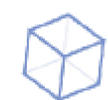
这里的 $\boldsymbol{o}_t$ 为 t 时刻的输出层上的状态。最后，经过输出门，保留在隐层上的成分为：

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) \tag{9.42}$$

综上所述，随着时间的变化，整个网络的结构设计流图见图 9.10，目前该网络已被成功地应用于手写识别、语音识别、机器翻译、图像或新闻标题生成与解析等。

9.4 典型应用

深度循环和递归神经网络在自然语言处理领域取得了诸多显著的成果，例如情感分析、机器翻译和问答系统等；和传统方法处理这些任务相比，深度循环和递归神经网络的重要特点是用向量表示各种级别的元素，传统方法会用很精细的方法去标注，而深度学习会用向量表示单词、短语、逻辑表达式和句子，然后通过搭建多层(引入定向循环)神经网络去自主学习。



9.4.1 深度循环神经网络的应用举例

深度循环神经网络已经被实践证明对自然语言处理是非常成功的，如语句合法性检查、词向量表达、词性标注等。在循环神经网络中，目前使用最广泛、最成功的模型便是长短时记忆神经网络，该模型通常比 Vanilla 循环神经网络能够更好地对长短时依赖进行表达，下面针对几种应用任务做以下简要分析。

1. 语言模型与文本生成

给你一个单词序列，需要根据前面的单词预测其后每一个单词出现的可能性。语言模型能够预测一个正确语句的可能性，这是机器翻译的一部分，往往可能性越大，语句越正确。另一种应用便是使用生成模型预测下一个单词的概率，从而生成新的文本根据输出概率的采样。在语言模型中，典型的输入是单词序列中每个单词的词向量，输出是预测的单词序列。当对网络进行训练时，如果有 $\boldsymbol{x}_{t+1}=\boldsymbol{o}_t$，那么 t 时刻的输出便是下一时刻的输入。为了下面的描述，首先给出词向量的定义。

定义 若词典里有 $|\boldsymbol{V}|$ 个词，每个词都被表示成一个 $|\boldsymbol{V}|$ 维的向量，设某个词在字典中相应的顺序为 i，则向量中 i 的位置上为 1，其余位置为 0。

问题描述 已知前 t 个时刻(包括 t 时刻)的历史数据，来预测 $t+1$ 时刻的输出数据，即

$$[\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_t]\xrightarrow{\text{预测}}\boldsymbol{x}_{t+1} \tag{9.43}$$

构建的模型如下：

$$\begin{cases}\boldsymbol{e}_t = L(\boldsymbol{D},x_t)\\ \boldsymbol{h}_t = \sigma(\boldsymbol{W}\cdot\boldsymbol{h}_{t-1}+\boldsymbol{U}\cdot\boldsymbol{e}_t+\boldsymbol{b})\\ \boldsymbol{y}_t = \mathrm{softmax}(\boldsymbol{V}\cdot\boldsymbol{h}_t+c)\end{cases} \tag{9.44}$$

其中 $L(\boldsymbol{D},\boldsymbol{x}_t)$是将第 t 时刻的词 $\boldsymbol{x}_t$ 通过词典 $\boldsymbol{D}$ 作用后得到的词向量，即根据词向量的定义，将此词向量记为循环神经网络的输入，其中 $\boldsymbol{y}_t$ 为输出，其对应的物理意义为预测下一时刻，即 $t+1$ 时刻的词向量，依据定义，便可以得到 $\boldsymbol{x}_{t+1}$。待优化学习的参数为 $\boldsymbol{U},\boldsymbol{W},\boldsymbol{V},\boldsymbol{b},\boldsymbol{c}$。优化目标函数为：

$$\min_{\theta} J(\theta) = -\sum_{i=1}^{|\boldsymbol{V}|} y_t^{(i)}\log(\hat{y}_t^{(i)}) \tag{9.45}$$

其中参数 $\theta=[\boldsymbol{U},\boldsymbol{W},\boldsymbol{V},\boldsymbol{b},\boldsymbol{c}]$，且：

$$\hat{y}_t^{(i)} = P(L(\boldsymbol{D},\boldsymbol{x}_{t+1}) = \boldsymbol{v}_i \mid \boldsymbol{x}_t,\cdots,\boldsymbol{x}_2,\boldsymbol{x}_1) \tag{9.46}$$

即在$[\boldsymbol{x}_1,\boldsymbol{x}_2,\cdots,\boldsymbol{x}_t]$已知的情形下，将 x_{t+1}所对应的词向量 $L(\boldsymbol{D},\boldsymbol{x}_{t+1})$预测为词字典中第 i 个位置的概率。求解仍使用梯度下降的方式，这里不再赘述。

备注：详细请参考博客 http://www.jianshu.com/p/b4c5ff7c450f.

2. 机器翻译

机器翻译是将一种源语言语句变成意思相同的另一种源语言语句，如将英语语句变成

同样意思的中文语句。与语言模型关键的区别在于：需要将源语言语句序列输入后，才进行输出，即输出第一个单词时，便需要从完整的输入序列中进行获取。这里仅给出一种简单的机器翻译结构图，如图 9.12 所示。

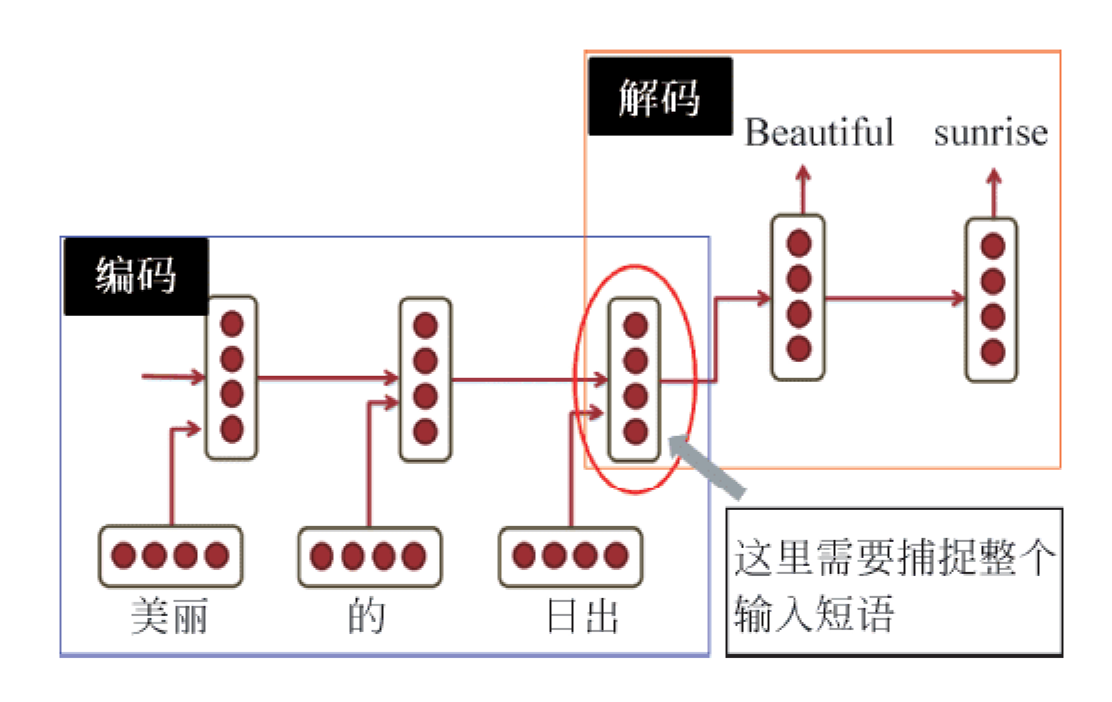


图 9.12　基于循环神经网络的机器翻译(从中文到英文)

备注：任务及物理解释请详细参考博客 http://www.jianshu.com/p/23b46605857e.

3. 语音识别

语音识别是指给一段声波的声音信号，预测该声波对应的某种指定源语言的语句以及该语句的概率值。

4. 图像描述生成

和卷积神经网络一样，循环神经网络已经在对无标注图像描述自动生成中得到应用，将卷积神经网络与循环神经网络结合进行图像描述自动生成；该组合模型能够根据图像的特征生成描述。

9.4.2　深度递归神经网络的应用举例

虽然深度循环和递归神经网络已经在语音和图像处理中取得重大进展，然而语言与语音、图像不同，是特殊的人工符号系统，将深度学习的方法应用于自然语言处理需要进行更多的研究和探索，针对特殊任务的词汇表达的学习以及词汇之间关系的探索越来越受到重视；处理自然语言的结构化输出需要更为复杂的循环神经网络；复杂神经网络又对高效和并行化的训练算法提出了新的要求。随着可用的训练数据越来越多，计算能力会越来越强，在自然语言处理领域，深度循环神经网络也会更有用武之地。

参考文献

[9.1]　Liwicki, Fernandez, Bertolami, et al. A Novel Connectionist System for Improved Unconstrained Handwriting Recognition[J]. Physics Letters B, 2008, 450(4): 332-338.

[9.2] Sak H, Senior A, Beaufays F. Long short-term memory recurrent neural network architectures for large scale acoustic modeling[J]. Computer Science, 2014: 338-342.

[9.3] Socher R, Lin C Y, Ng A Y, et al. Parsing Natural Scenes and Natural Language with Recursive Neural Networks[C]. International Conference on Machine Learning, ICML 2011, Bellevue, Washington, Usa, June 28-July. DBLP, 2011: 129-136.

[9.4] Socher R, Perelygin A, Wu J Y, et al. Recursive deep models for semantic compositionality over a sentiment treebank[J]. 2013.

[9.5] Cruse H. Neural Networks as Cybernetic Systems [M]. Thieme Medical Publishers, Incorporated, 1996.

[9.6] Jaeger H, Haas H. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication[J]. Science, 2004, 304(5667): 78.

[9.7] Hochreiter S. Untersuchungen zu dynamischen neuronalen Netzen[C]. Master's Thesis, Institut Fur Informatik, Technische Universitat, Munchen. 1991.

[9.8] Schmidhuber J. Learning Complex, Extended Sequences Using the Principle of History Compression[J]. Neural Computation, 1992, 4(2): 234-242.

[9.9] Schmidhuber J. Deep Learning[J]. Scholarpedia, 2016, 10(11).

[9.10] Bayer J,Wierstra D, Togelius J, et al. Evolving Memory Cell Structures for Sequence Learning [C]. International Conference on Artificial Neural Networks. Springer-Verlag, 2009: 755-764.

[9.11] Fernández S, Graves A, Schmidhuber J. Sequence Labelling in Structured Domains with Hierarchical Recurrent Neural Networks[C]. IJCAI 2007, Proceedings of the International Joint Conference on Artificial Intelligence, Hyderabad, India, January. DBLP, 2007: 774-779.

[9.12] Graves A,Ndez S, Gomez F, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks[C]. International Conference on Machine Learning. ACM, 2006: 369-376.

[9.13] Ndez S, Graves A, Schmidhuber J, et al. An application of recurrent neural networks to discriminative keyword spotting[C]. Artificial Neural Networks-ICANN 2007, International Conference, Porto, Portugal, September 9-13, 2007, Proceedings. DBLP, 2007: 220-229.

[9.14] Hannun A, Case C, Casper J, et al. Deep Speech: Scaling up end-to-end speech recognition[J]. Computer Science, 2014.

[9.15] Li X, Wu X. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition[C]. IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2014: 4520-4524.

[9.16] Fan B, Wang L, Soong F, et al. Photo-Real Talking Head with Deep Bidirectional LSTM [J]. 2015.

[9.17] Zen H,Sak H. Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis[C]. IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2015: 4470-4474.

[9.18] Gers F A, Schmidhuber E. LSTM recurrent networks learn simple context-free and context-sensitive languages[J]. IEEE Transactions on Neural Networks, 2001, 12(6): 1333-1340.

[9.19] Graves A, Schmidhuber J, rgen. 2005 Special Issue: Framewise phoneme classification with bidirectional LSTM and other neural network architectures[J]. Neural Networks, 2005, 18(5-6): 602-610.

[9.20] Abboud J M, Balázs N, Jean-Claude G, et al. Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription[J]. Chemistry A European Journal, 2012, 18(13): 3981-3991.

[9.21] Socher R, Lin C Y, Ng A Y, et al. Parsing Natural Scenes and Natural Language with Recursive Neural Networks[C]. International Conference on Machine Learning, ICML 2011, Bellevue, Washington, Usa, June 28-July. DBLP, 2011: 129-136.

[9.22] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model[C]. INTERSPEECH 2010, Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September. DBLP, 2010: 1045-1048.

[9.23] Sutskever I, Martens J, Hinton G E. Generating Text with Recurrent Neural Networks[C]. International Conference on Machine Learning, ICML 2011, Bellevue, Washington, Usa, June 28-July. DBLP, 2011: 1017-1024.

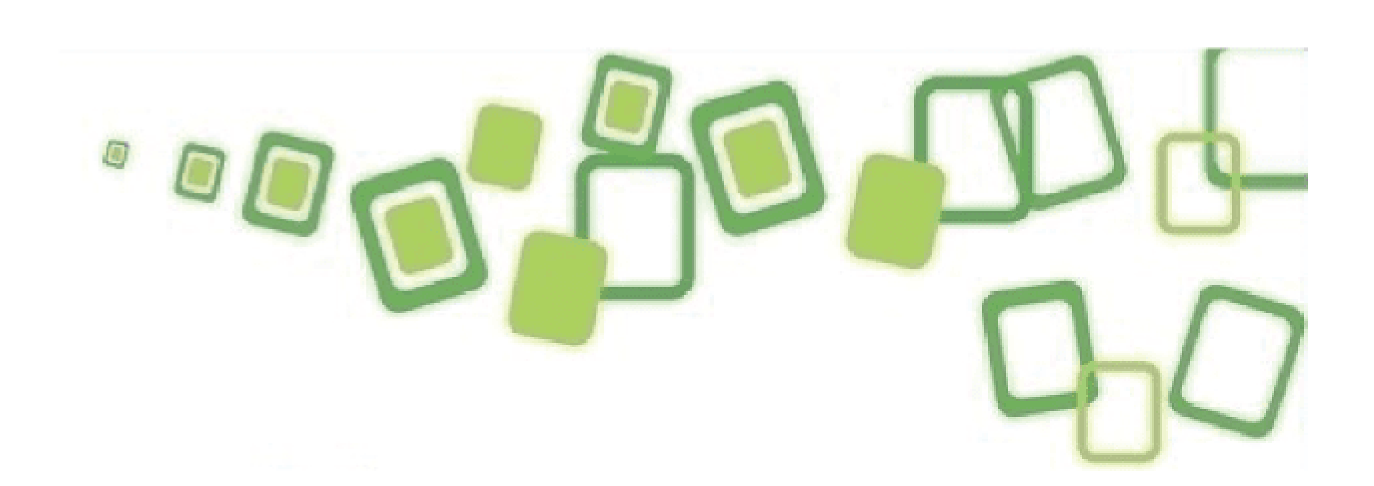
[9.24] Sutskever I, Vinyals O, Le Q V. Sequence to Sequence Learning with Neural Networks[J]. Advances in Neural Information Processing Systems, 2014, 4: 3104-3112.

[9.25] Auli M, Galley M, Quirk C, et al. Joint language and translation modeling with recurrent neural networks[J]. American Journal of Psychoanalysis, 2013, 74(2): 212-213.

[9.26] Graves A, Jaitly N. Towards end-to-end speech recognition with recurrent neural networks[C]. International Conference on Machine Learning. 2014: 1764-1772.

[9.27] Liu S, Yang N, Li M, et al. A Recursive Recurrent Neural Network for Statistical Machine Translation[C]. Meeting of the Association for Computational Linguistics. 2014: 1491-1500.

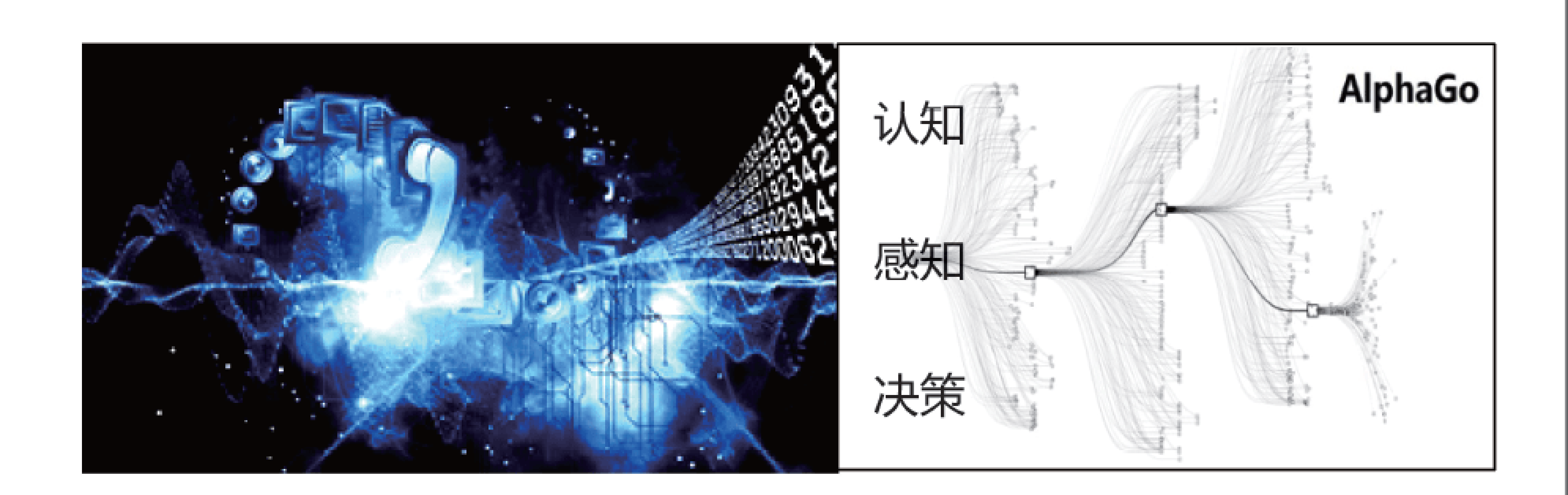
第10章



深度强化学习

CHAPTER 10

人工智能——深度学习+强化学习



深度学习导致感知方面巨大突破

强化学习引起决策方面巨大突破



10.1 深度强化学习简介

10.1.1 深度强化学习的基本思路

深度强化学习将深度学习的感知能力和强化学习的决策能力相结合，可以直接根据输入的图像进行控制，是一种更接近人类思维方式的人工智能方法。众所周知，在人工智能领域，感知、认知和决策的能力都是衡量智能的指标，可以认为：深度学习（深度神经网络）是使得感知能力得到进一步提升与巨大突破的核心技术，同时，强化学习的学习机制表明它是不断地与环境交互（可以看作决策系统和环境的博弈），以试错的学习方式得到最优策略，是使得决策能力持续获取收益的关键技术。深度强化学习的原理框架如图10.1所示。

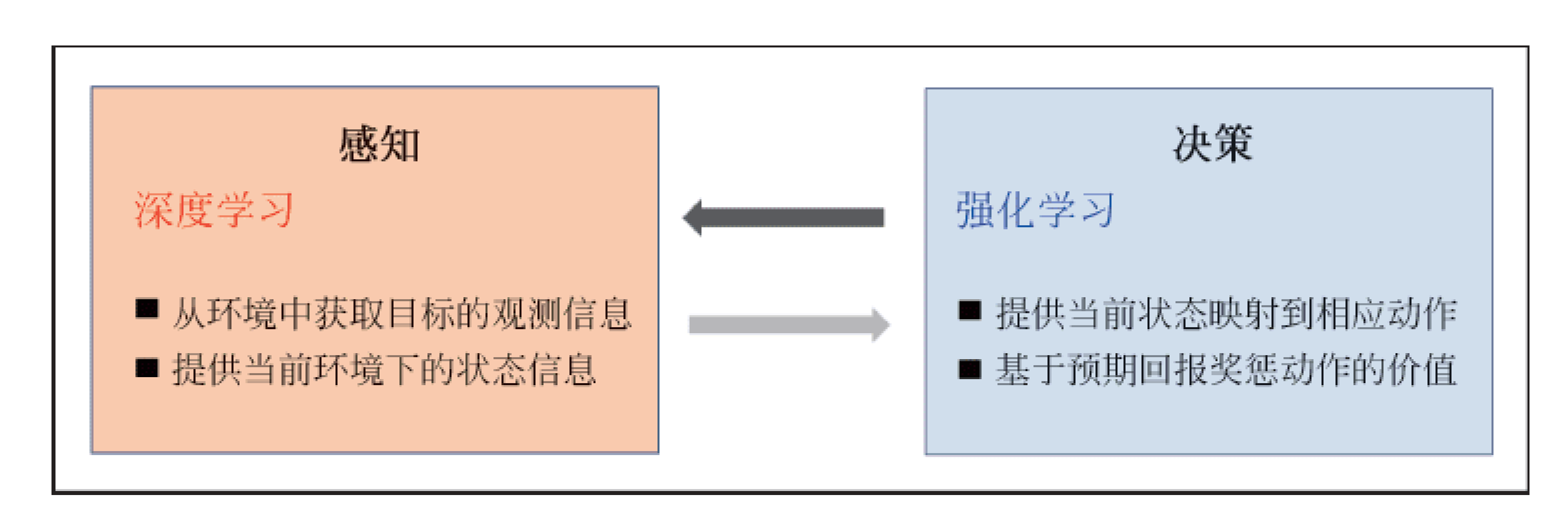


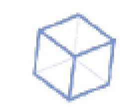
图10.1 深度强化学习的框架

另外，已知大脑的信息加工能力是有限的，不可能在瞬间进行多种操作，为了顺利地加工大量的信息，人只能按照一定的策略在每一时刻选择特定的信息进行操作，并将整个认知（包括感觉、知觉、记忆、思维、想象、言语等）过程的大量操作组织起来。因此，感知、认知和决策对于认知活动的有效进行与展开是十分重要的。

备注：强化学习是受到生物能够有效适应环境的启发，以试错的机制与环境进行交互，通过最大化累积奖赏的方式来学习最优策略，简言之，最简单的理解就是在训练的过程中，不断地去尝试，错了就惩罚，对了就奖励，由此训练得到在各个状态环境当中最好的决策，例如骑车的过程、种瓜的过程等。具体地，可以参考第1.3节关于强化学习的数学分析。

10.1.2 发展历程

谷歌的DeepMind团队在Nature杂志上发表的两篇文章（基于视频游戏的深度强化学习算法和AlphaGo围棋程序）使得深度强化学习成为高级人工智能的热点。在此之前，已出现了一些类似的研究工作，它们的主要思路是利用神经网络将复杂高维的数据降维，转化到低维特征空间便于强化学习处理，例如Shibata等将浅层神经网络和强化学习结合起来处理视觉信号的输入，控制机器人完成推箱子等游戏；又如Lange等人提出将深度自编码



器应用到视觉的学习控制中，提出了视觉动作学习，使智能体具有感知和决策能力；随后，Abtahi 等将深度置信网络引入到强化学习中，将传统的值函数利用深度置信网络来替代，并将其成功地应用在车牌图像的字符分割任务上；还有，Lange 进一步将视觉输入的强化学习应用到车辆控制中，该框架被称为深度拟合 Q 学习（所谓 Q 学习是指状态-动作值函数学习）。之前，强化学习不能实用的主要原因在于面对过大的状态或者行动空间，很难有效地处理这些情形。深度学习的出现能够去处理这些情形背后的真正问题，如 ImageNet 数据集上视觉识别准确率的大幅提高，即 top5 错误率下降到 4%以内，深度学习相关技术已在图像和语音识别领域变得比较成熟并且已被广泛商用。以上说明深度学习已成为一些实际应用的基础，而深度强化学习的研究及应用也基本上按照上面的思路展开。下面简要给出强化学习和深度学习的研究历程。

1. 强化学习简要发展历程

(1) 1956 年 Bellman 提出了动态规划方法。

(2) 1977 年 Werbos 提出了自适应动态规划方法。

(3) 1988 年 Sutton 提出了 TD 算法。

(4) 1992 年 Watkins 提出了 Q(状态-动作值函数)学习算法。

(5) 1999 年 Thrun 提出了部分可观测马尔科夫决策过程中的蒙特卡洛方法。

(6) 2006 年 Kocsis 提出了置信上限树算法。

(7) 2014 年 Silver 等提出了确定性策略梯度算法。

2. 深度学习简要发展历程

(1) 1974 年 Werbos 提出了 BP 算法。

(2) 1986 年 Rumelhart 等人重新发明 BP 算法，BP 算法的实质是最小均方算法的推广。

(3) 1995 年和 1998 年，LeCun 和 Bengio 等人提出并改进了卷积神经网络。

(4) 2006 年 Hinton 提出了逐层预训练方法，解决梯度弥散问题的深度置信网络。

(5) 2008 年 Vincent 等提出了降噪自编码器。

(6) 2011 年 Rafir 等提出了收缩自编码器。

(7) 2012 年 Krizhebsky 提出了 AlexNet 网络，并在 ImageNet 数据集上取得突破。

(8) 2014 年 Ian Goodfellow 提出了生成式对抗网络。

(9) 2015 年何铠明等提出了深度残差网络。

将深度学习与强化学习相结合，已在理论和应用方面取得了显著的成果，特别是谷歌的 DeepMind 团队研发的围棋程序 AlphaGo 及其升级版 Master，在 2016 年以 4∶1 的比分战胜九段围棋选手李世石，成为人工智能历史上又一个新的里程碑。另外深度强化学习在博弈均衡求解中的应用也是令人兴奋的方向之一，随着这些技术的细化和深入，进一步将理论计算机和更为实用的机器学习等技术之间的鸿沟缩小。随着理解的不断深入，将会发现深

度强化学习考量各个应用领域有趣的问题并放置在同一个框架内进行思考和处理，逐步探索这些有趣的问题，最终能够取得满意的结果，从而实现框架和模型的重要性在于可以将抽象的概念和理论转化为触手可及的经验。

10.1.3 应用新方向

深度强化学习是近两年来深度学习领域迅猛发展起来的一个分支，其目的是解决计算机从感知到决策控制的问题，从而实现通用人工智能（通用人工智能是要创造一种无须人工编程，自学习解决各种问题的智能体，最终实现类人级别甚至超人级别的智能，通用的人工智能见图 10.2）。目前，以谷歌的 DeepMind 团队为首，深度强化学习已在视频、游戏、围棋和机器人等领域取得了突破性的进展，例如 AlphaGo，其核心在于使用了深度强化学习，使得计算机能够通过自对弈的方式不断地提升棋艺水平，值得指出的是从感知到决策，端到端设计模式的深度强化学习具有非常广阔的应用前景，它的发展将进一步推动高级人工智能的革命。

其中智能体是指任何独立的能够思想并可以同环境交互的实体都可以抽象为智能体，这里特指以深度强化学习为核心的技术框架，其中深度学习用来提供学习的机制，强化学习为深度学习提供学习的目标。目前，以深度强化学习为核心的基本框架都可以容纳在行动和评判模块下，如图 10.3 所示。

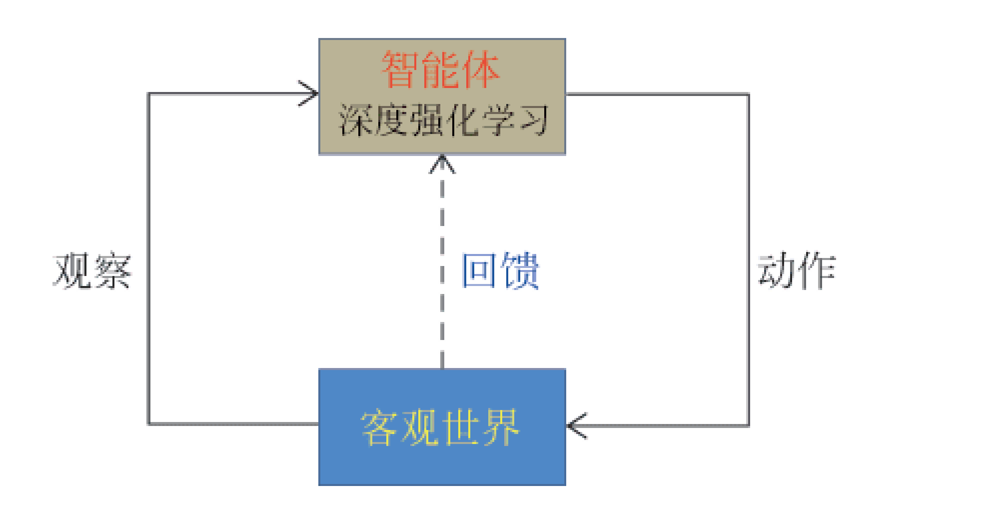


图 10.2 通过人工智能的基本框架

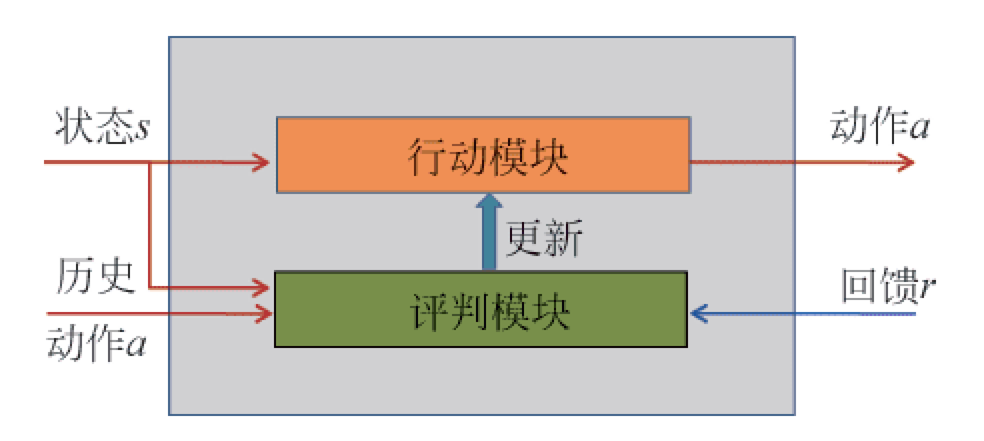
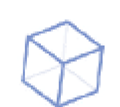


图 10.3 行动-评判框架

若将深度强化学习比作智能体中的大脑，那么该大脑包括两个模块：行动模块和评判模块，其中行动模块是大脑的执行机构，通过输入外部的状态 s，然后输出动作 a，而评判模块则可以认为是大脑的价值观，根据历史的信息和回馈 r 进行更新，即自我调整然后影响整个行动模块。注意人类也是在自身价值和本能的指导下进行行为，并且价值观受经验的影响不断改变。

备注：在行动-评判框架下，谷歌的 DeepMind 相继提出了深度 Q 网络（DQN，于 2013 年提出），A3C（Asynchronous Advantage Actor Critic，于 2015 年提出）和 UNREAL（Unsupervised Reinforcement and Auxiliary Learning，于 2016 年 11 月提出）等三种深度强化学习算法。

深度强化学习经过近两年的发展，在算法层面上取得了越来越好的效果，精妙的算法设



计无不闪耀着人工智慧的光芒，作为从感知到决策控制的通用学习算法，已在各个领域得到广泛的应用。

10.2　深度Q网络

Q学习是1989年Watkins提出的，是最早的在线强化学习算法，同时也是强化学习最重要的算法之一。而深度Q网络(Deep Q Network)是谷歌的DeepMind于2013年提出的第一个深度强化学习算法，并在2015年进一步完善，成果发表在Nature杂志上。DeepMind将深度Q网络应用在计算机玩Atari游戏上，不同于以往的处理方式，仅使用视频信息作为输入，与人类玩游戏一样。注意：符号“Q”表示在某一状态下执行某一操作时所获取的分数或质量。

10.2.1　网络基本模型与框架

在深度Q网络中，仅用值网络来表示评判模块，没有使用行动模块，因为使用评判模块即可选择并执行最优的动作。其核心思想是：基于值网络，可以遍历某个状态下各种动作的价值，然后选择价值最大的一个动作输出，如图10.4所示。

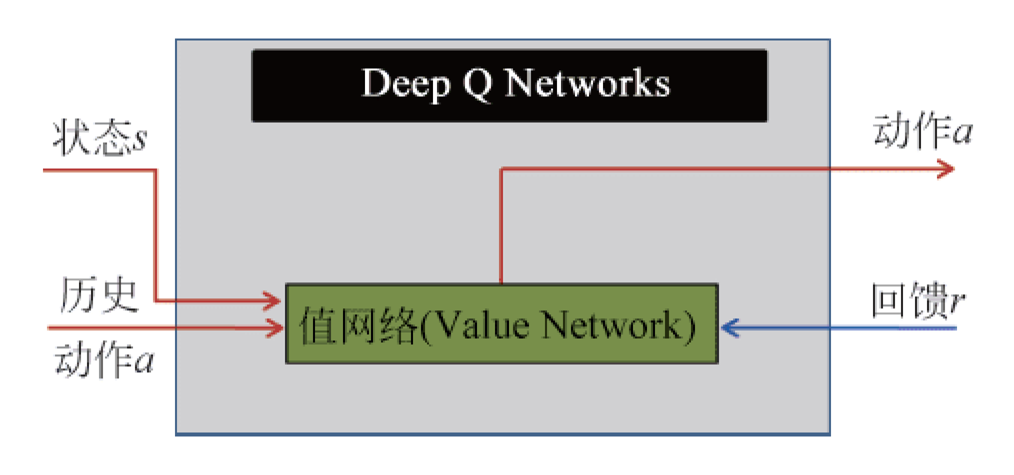


图10.4　深度Q网络的基本框架

注意深度Q网络作为深度强化学习的第一个算法，仅使用价值网络，其特点是训练效率低并且只能面向低维的离散控制问题，换言之，通用性有限。但又由于它是第一次成功结合深度学习和增强学习，解决了高维数据输入问题，并且在Atari游戏上取得重大突破，具有开创性的意义。由于深度卷积神经网络在图像处理领域有着天然的优势，将其与强化学习中的Q学习相结合处理图像数据的感知决策任务成为目前的主流方向之一；例如采用时间上相邻的4帧游戏画面作为原始图像输入，经过深度卷积神经网络处理，网络的输出为状态和行动下的函数值，实现端到端方式下的学习控制。

备注：斯坦福大学在线Demo展示了深度Q网络的性能及实现，参考地址：http://cs.stanford.edu/people/karpathy/convnetjs/demo/rldemo.html，见图10.5。

注意：该游戏中，吃到绿色的(毒药)，惩罚得负分；吃到红色的(苹果)，奖赏得分；根据速度的设置，它具有超速、快速、正常和慢速四个等级满足不同状态下对行为的反馈。

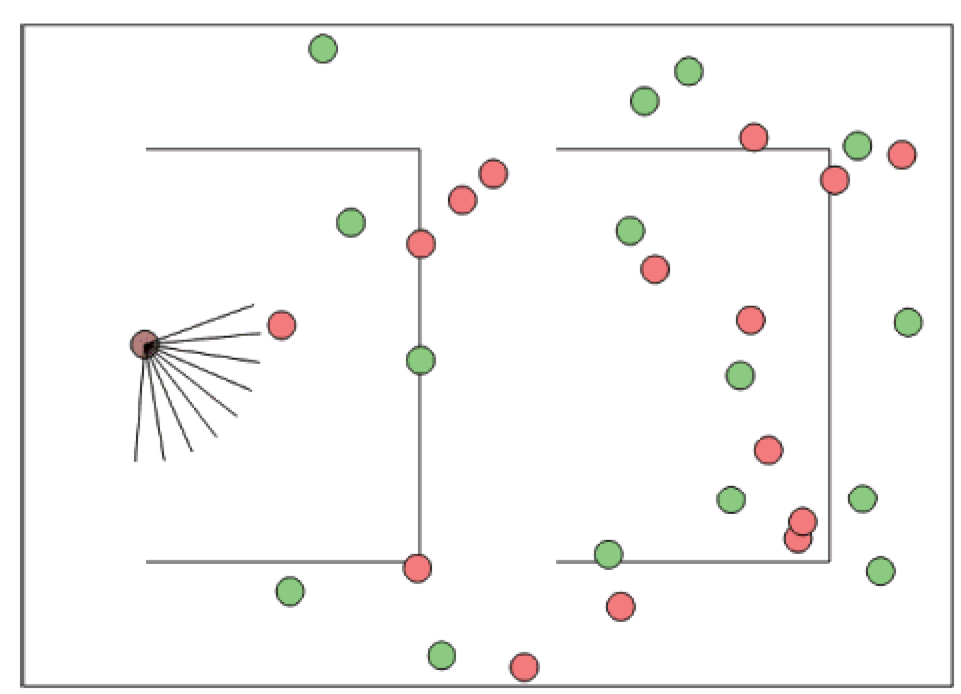


图 10.5　基于深度强化学习(卷积神经网络+Q学习)实现游戏

10.2.2　深度Q网络的数学分析

1. Q学习的分析

通常某一游戏,例如象棋、围棋等,可将其拆分为一系列的步骤(最后一步为胜负结果),其中每一步都包括竞技双方根据当前自己的观测所做的行动或行为,同时期望该行为所带来的累积奖赏(即该行为及之前行为的奖赏和)优于对方,为自己最终获胜增添筹码。如何利用数学分析来量化这一过程?首先,必要的假设为:游戏中每一步所对应的观测,可执行的行动或行为是有限的;不可能仅靠当前这一步的观测来执行某一行为,需要充分结合该步之前的观测和行为来采取行动。

为了下面描述的方便,给出几个名词描述:

- 符号 $\boldsymbol{x}_t \in \mathbb{R}^{m\times n}$ 为游戏进行到第 $t(t=1,2,\cdots,T)$ 步时,所对应的观测(图像)。
- 符号 $a_t \in \Lambda$ 为观测 $\boldsymbol{x}_t$ 下所执行的动作,其中 Λ 为游戏规则下合理行动集合。
- 符号 $\boldsymbol{r}_t$ 为观测 $\boldsymbol{x}_t$ 下执行动作 a_t 后,所获取的奖赏(或惩罚),另外:

$$R_t = \sum_{t'=t}^{T} \gamma^{(t'-t)} \cdot \boldsymbol{r}_{t'} \tag{10.1}$$

其中 $\gamma \in (0,1)$ 为折扣因子,这里的 R_t 为第 t 步到终止时刻 T 所获取的累积奖赏和。

- 符号 $Q(s,a)$ 为状态动作值函数,其中 t 时刻的状态 s 为:

$$s_t = (\boldsymbol{x}_1, a_1, \cdots, \boldsymbol{x}_{t-1}, a_{t-1}, \boldsymbol{x}_t) \tag{10.2}$$

接下来,给出Q学习的主要思路,即根据如下的迭代公式实现状态动作值函数的优化学习:

$$\begin{cases} Q_{k+1}(s_t, a_t) = Q_k(s_t, a_t) + \alpha_k \cdot \boldsymbol{\delta}_k \\ \boldsymbol{\delta}_k = \boldsymbol{r}_{t+1} + \gamma \cdot \max\limits_{a' \in \Lambda} Q_k(s_{t+1}, a') - Q_k(s_t, a_t) \end{cases} \tag{10.3}$$

其中 α_k 为学习速率,另外 s_t 和 a_t 分别为第 t 步所对应的状态和行动,$\boldsymbol{\delta}_k$ 为时间差分,γ 为折扣因子,a' 为 Λ 中使得第 k 次迭代下的状态动作值函数在 s_{t+1} 下可执行的动作。已经证明

当式(10.3)满足以下两个条件时,一是:

$$\sum_k \alpha_k^2 < +\infty, \quad \sum_k \alpha_k = +\infty \tag{10.4}$$

二是所有的状态动作都能被无限次地遍历,则有:

$$\lim_{k \to \infty} Q_k = Q^* \tag{10.5}$$

即随着迭代次数趋于无穷时得到最优控制策略下的状态动作值函数,类似种西瓜任务,总结出了一套从播种到结果其中每一个状态下的最优执行策略。

最后,依据(Bellman equation)上面的描述,可知某一状态下选择最优可执行动作的策略,便是最大化如下期望值:

$$Q^*(s,a) = \mathbb{E}_{s' \sim \xi}(\boldsymbol{r} + \gamma \cdot \max_{a'} Q^*(s',a') \mid s,a) \tag{10.6}$$

其中 ξ 为环境,所有的状态都包含在内,另外 s' 分别为状态 s 执行动作 a 之后的状态,a' 为符合游戏规则下针对状态 s' 所有可能执行的动作。

注意式(10.6)采用的是期望来分析状态动作值函数,在实际应用中,常使用函数逼近的策略实现状态动作值函数的估计,即

$$Q(s,a,\theta) \approx Q^*(s,a) \tag{10.7}$$

在强化学习中,常用的逼近方式有线性的和非线性的(例如神经网络),注意待优化的权值参数为 θ。假设对于每一次迭代 k,Q 网络通过最小化如下的目标函数来实现参数更新:

$$L_k(\theta_k) = \mathbb{E}_{s,a \sim \rho(\cdot)}[(\boldsymbol{y}_k - Q(s,a;\theta_k))^2] \tag{10.8}$$

这里的 $\rho(\cdot)$ 为行为分布,即 $\rho(s,a)$ 为状态 s 和行为 a 的概率分布,另外 $\boldsymbol{y}_k$ 为第 k 次迭代所对应的目标(输出),且有:

$$\boldsymbol{y}_k = \mathbb{E}_{s' \sim \xi}[\boldsymbol{r} + \gamma \cdot \max_{a'} Q(s',a';\theta_{k-1}) \mid s,a] \tag{10.9}$$

依据式(10.8)和式(10.9)有:

$$\theta_0 \xrightarrow{\min_\theta L_1(\theta_1)} \theta_1 \xrightarrow{\min_\theta L_2(\theta_2)} \cdots \xrightarrow{\min_\theta L_k(\theta_k)} \theta_k \rightarrow \cdots \tag{10.10}$$

即依据式(10.9),在参数 θ_0 已知的前提下,可以得到目标输出 $\boldsymbol{y}_1$,再通过优化目标函数式(10.8)更新参数,得到 θ_1,以此类推,最终实现参数的收敛,即

$$\lim_{k \to \infty} \theta_k = \theta_* \tag{10.11}$$

其中对于式(10.8)参数的更新求解采用梯度下降法,其中偏导数为:

$$?_{\theta_k} L_k(\theta_k) = \mathbb{E}_{s,a \sim \rho(\cdot); s' \sim \xi}[(\boldsymbol{r} + \gamma \cdot \max_{a'} Q(s',a';\theta_{k-1}) - Q(s,a;\theta_k)) \cdot ?_{\theta_k} Q(s,a;\theta_k)] \tag{10.12}$$

注意这里参数 θ_{k-1} 是固定的。

2. 深度 Q 学习的分析

目前,依托大量训练数据集而成功的深度学习技术已在计算机视觉和语音处理等领域取得诸多突破性成果,最为直观的结论便是:依赖数据中的先验知识挖掘统计或物理特性的特征工程(包括特征提取与体征选择)将被基于深度学习技术下的特征学习所替代。

备注:特征学习与特征工程分别是深度学习和机器学习下挖掘数据中所蕴含的某种语

义或统计特性，二者对数据的规模要求不同，且处理技术中蕴含的思想也不一样。

通常Q学习技术的成功依赖于人工特征的选取，进一步，智能体学习的好坏严重地取决于特征选取的质量。能否将Q学习中的人工特征提取技术替换为深度学习下的特征学习，如基于卷积神经网络的特征学习？回答是肯定的，这便是深度Q学习的动机。下面便简要地给出深度Q学习的数学分析，首先，介绍一个概念，即经验回放(Experience Replay)，它存储着某一智能体(即游戏)在任意一步的经验，如：

$$\boldsymbol{e}_t = (s_t, a_t, \boldsymbol{r}_t, s_{t+1}) \tag{10.13}$$

其中出现的符号与Q学习中的一致，$\boldsymbol{e}_t$ 为第 t 步的经验，假设该智能体结束时的步数为 N，那么经验回放便可对应着集合：

$$\boldsymbol{D} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_N] \tag{10.14}$$

其次，对于Q学习中状态动作值函数改为：

$$Q(s, a, \theta) \xrightarrow{\text{修正}} Q(\varphi(s), a, \theta) \tag{10.15}$$

这里 $\varphi(\cdot)$ 为基于深度学习下的特征学习，注意两点，一是这里的参数 θ 不仅包含之前Q学习中的参数，而且还有深度学习下的参数；二是对于Q学习中的第 t 步状态有：

$$s_{t+1} = (s_t, a_t, \boldsymbol{x}_{t+1}) \tag{10.16}$$

其中 $s_1 = [\boldsymbol{x}_1]$，且 $t = 1, 2, \cdots, T$。而式(10.15)，左边对应Q学习下的，其状态输入为 s，右侧对应深度Q学习下的，其"状态"输入为 $\varphi(s)$，即此时的经验回放对应修正为：

$$\begin{cases} \boldsymbol{D} \xrightarrow{\text{修正}} \bar{\boldsymbol{D}} = [\bar{\boldsymbol{e}}_1, \bar{\boldsymbol{e}}_2, \cdots, \bar{\boldsymbol{e}}_N] \\ \bar{\boldsymbol{e}}_t = (\varphi(s_t), a_t, \boldsymbol{r}_t, \varphi(s_{t+1})) \end{cases} \tag{10.17}$$

再次，优化目标函数所对应的式(10.8)和期望输出 $\boldsymbol{y}_k$ 对应的式(10.9)，其中的状态动作值函数的输入状态 s 应修正为 $\varphi(s)$。最后，求解仍沿用梯度下降法。

10.3 应用举例——AlphaGo

当前，基于深度强化学习的AlphaGo(计算机围棋程序)已经成为人工智能领域的焦点。一直以来，计算机围棋被认为是人工智能领域的一大挑战，本质上是因为大约要搜索 b^d 个落子情况序列，其中 b 为搜索的宽度(即当前局面在哪里落子)，d 为搜索的深度(即当前局面在接下来若干步之后的对弈局面)，以期望利用状态动作值函数来评估当前棋局和落子的最佳位置。与象棋等具有有限且可执行的搜索空间不同，围棋的计算复杂度约为 250^{150}，如果按照现有的计算能力采用暴力的搜索方式是不能解决问题的。早期的计算机围棋通过专家系统和模糊匹配缩小搜索空间，减轻计算强度，但由于计算能力有限，取得的实际效果并不理想。近些年，随着深度学习的不断发展和完善，基于深度强化学习和蒙特卡罗树搜索策略的计算机围棋程序AlphaGo已达到人类顶尖棋手的水准，其核心思路是通过卷积神经网络来构建的价值网络和策略网络分别对搜索的深度和宽度进行约减，使得搜索效



率大幅度提升，胜率估算也更加精确，如图 10.6 所示。

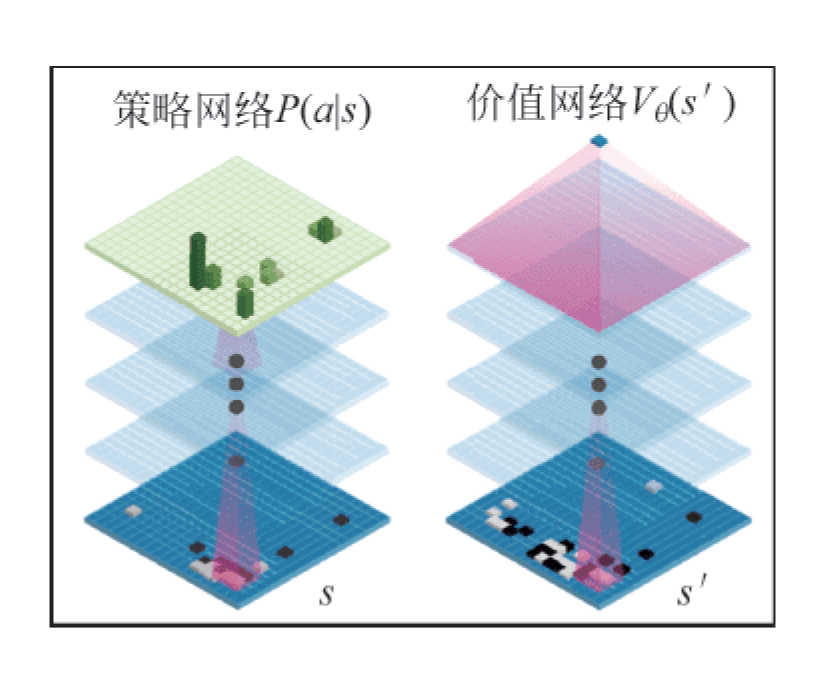


图 10.6　策略网络与价值网络——从广度与深度约减解空间

图 10.6 中的策略网络将当前棋盘状态 s 作为输入，经过多层的深度卷积神经网络输出不同落子位置的概率 $P(a|s)$，网络的优化训练可通过监督学习方式下的深度强化学习实现；价值网络同样使用深度卷积神经网络，输出一个标量值 $V_\theta(s')$来预测选择落子位置 s'时的累积奖赏，注意 s'表示当前状态 s 在执行动作 a 之后的状态，另外 θ 为价值网络的参数。

10.3.1　AlphaGo 原理分析

AlphaGo 操作原理流程图如图 10.7 所示。

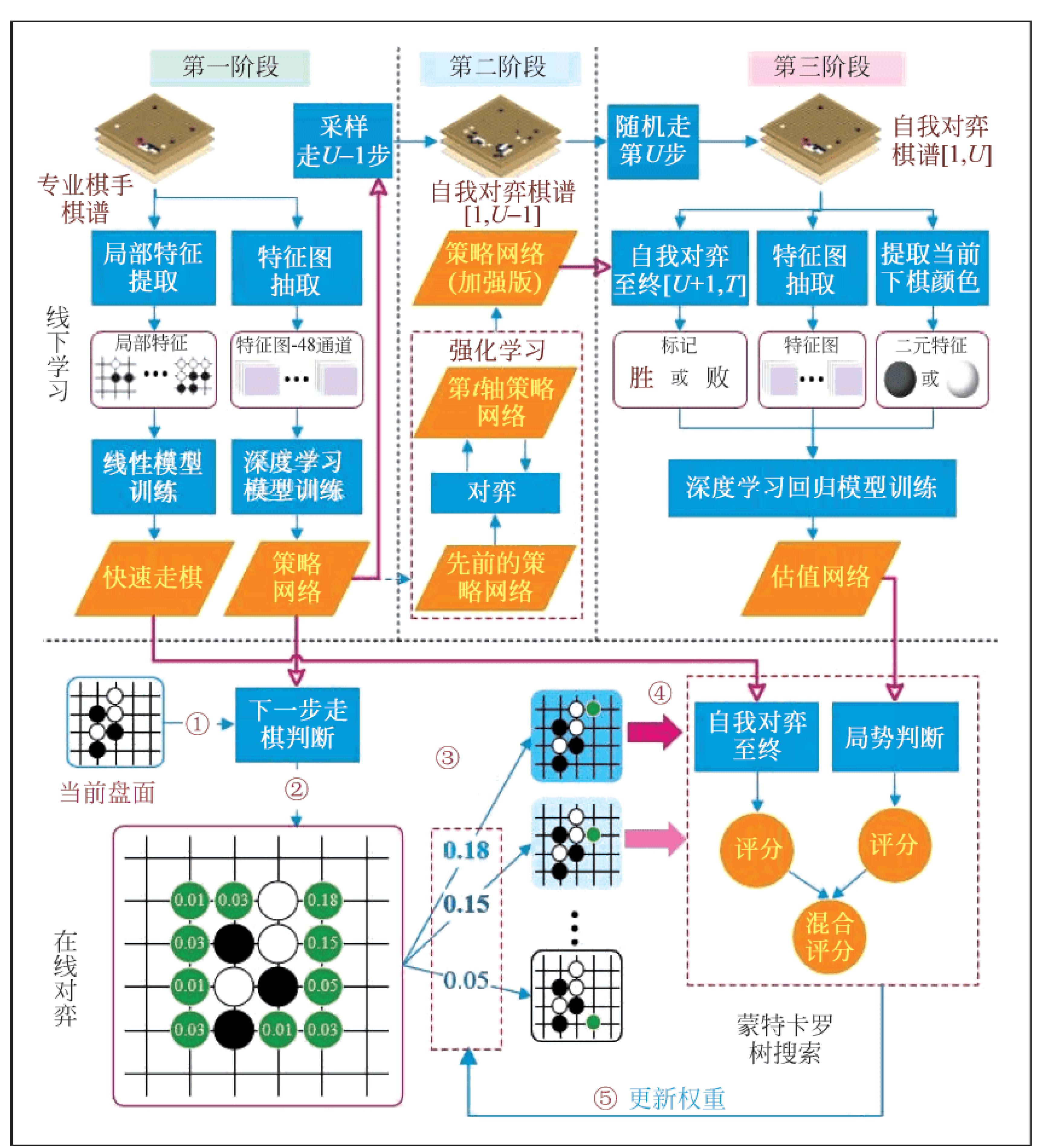


图 10.7　AlphaGo 操作原理流程图

备注：该原理图参考链接为 http://www.kddchina.org/#/Content/alphago

整体上，AlphaGo 的实现包括线下学习和在线对弈，其中线下学习包括三个阶段：

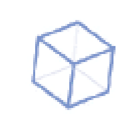
第一阶段：核心利用大量专业棋手的棋谱训练两个网络，即策略网络和快速走棋网络，其中策略网络采用深度卷积神经网络来训练学习。

第二阶段：基于强化学习来提升策略网络的性能，也可认为是围棋程序的自我对弈学习。

第三阶段：通过大量的自我对弈，实现基于深度强化学习的价值网络学习。

另外，在线对弈包括 5 个关键的步骤：

步骤一：依据当前对弈盘面进行特征提取，注意这里并不使用深度学习来提取特征，而



是将提取后的特征图作为深度学习的输入。

步骤二：依据策略网络估计棋盘其他空地的落子概率(搜索范围为宽度)。

步骤三：依据落子的概率，计算此处往下发展的权重，其中初始值为落子概率本身。

步骤四：利用价值网络和快速走棋网络分别判断局势，两个局势得分相加为此处走棋获胜的得分。

步骤五：利用蒙特卡罗树搜索展开下一步的搜索(搜索范围为深度)，并更新权重。

目前依据网络原理和实战经验可知，AlphaGo 相对人类的优势在于它的大局观天生比人强得多，因为有强大的计算资源保证模拟的终局数量足够，策略网络和价值网络剪枝又保证了模拟的质量。但是，AlphaGo 也存在以下三方面的缺陷：

- 打劫问题分析：AlphaGo 会尽量避免打劫。原因是打劫会导致后续算法变得异常复杂，使得结果充满不确定性。
- 策略网络存在隐患：基于历史棋局、自我对弈的强化学习对可能下棋的点做出概率判断，从而缩小落子范围。但通常由于每步落子时间的限制，导致放弃精确搜索，可能会存在将某些重大隐患但概率较小的落子位置忽略。
- 价值网络也有概率隐患：价值网络，即预测每一步及其后续步骤构成的赢棋概率。同样因为预测步骤有限，有时会对一些目前可行性较小、明显对自己有利的棋之后续较长远局势缺乏正确判断。

总之，针对 AlphaGo，行之有效的策略便是打劫要趁早，太晚了将导致搜索空间变小，即便价值网络失效，还可以靠快速走棋网络来弥补。开劫应该以在刚刚进入中盘时期为好，并且保持长时间不消劫，最好在盘面上能同时有两处以上打劫。

10.3.2　深度强化学习性能分析

深度强化学习是将深度学习和强化学习各自的优势进行组合后形成的一种接近人类思维方式的人工智能方法，除了将深度卷积神经网络与 Q 学习结合得到深度 Q 网络外，还有将深度递归神经网络和 Q 学习结合形成的深度递归 Q 网络，主要用于处理文本游戏，其中的深度递归神经网络可以将文本信息映射到向量空间从而获取游戏状态的语义信息。需要指出的是：强化学习的本质为一马尔科夫决策过程，它与机器学习中的监督学习不一样，强化学习不给定输入所对应的标注，而是给出一个回报函数，即决定在某状态下执行某动作的风险(有可能是好的结果，也有可能是坏的结果)。然而，强化学习性能的优劣取决于人工特征提取技术，深度学习的优势恰好可以弥补此短板。目前，主流的深度强化学习都是针对离散状态和动作的优化问题，而实际应用中状态和动作常是连续的，针对连续状态和动作的情形研究较少，从而也限制了深度强化学习应用的范围。应该清楚地看到，深度强化学习所带来的技术变革将持续在我们今后日常生活中产生广泛且有力的冲击响应，如智能驾驶、智能医疗、个人手机助手、智能机器人和无人航行器、智能制造等。

参考文献

[10.1] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529-533.

[10.2] Littman M L. Reinforcement learning improves behaviour from evaluative feedback[J]. Nature, 2015, 521(7553): 445-451.

[10.3] Lewis F L, Vrabie D. Reinforcement learning and adaptive dynamic programming for feedback control[J]. IEEE Circuits & Systems Magazine, 2009, 9(3): 32-50.

[10.4] Wei Q, Liu D. A New Discrete-Time Iterative Adaptive Dynamic Programming Algorithm Based on Q-Learning[M]. Advances in Neural Networks-ISNN 2015. Springer International Publishing, 2015.

[10.5] Mnih V, Kavukcuoglu K, Silver D, et al. Playing Atari with Deep Reinforcement Learning[J]. Computer Science, 2013.

[10.6] Koutník J, Schmidhuber J, Gomez F. Online Evolution of Deep Convolutional Network for Vision-Based Reinforcement Learning[M]. From Animals to Animats 13. 2014: 260-269.

[10.7] Lin L J. Reinforcement learning for robots using neural networks[J]. 1993.

[10.8] Nair A, Srinivasan P, Blackwell S, et al. Massively Parallel Methods for Deep Reinforcement Learning[J]. Computer Science, 2015.

[10.9] Guo X, Singh S, Lee H, et al. Deep learning for real-time Atari game play using offline Monte-Carlo tree search planning [C]. International Conference on Neural Information Processing Systems. MIT Press, 2014: 3338-3346.

[10.10] Hasselt H V, Guez A, Silver D. Deep Reinforcement Learning with Double Q-learning[J]. Computer Science, 2015.

[10.11] Osband I, Blundell C, Pritzel A, et al. Deep Exploration via Bootstrapped DQN[J]. 2016.

[10.12] Mnih V, Badia A P, Mirza M, et al. Asynchronous Methods for Deep Reinforcement Learning [J]. 2016.

[10.13] Sorokin I, Seleznev A, Pavlov M, et al. Deep Attention Recurrent Q-Network[J]. Computer Science, 2015.

[10.14] Lillicrap T P, Hunt J J, Pritzel A, et al. Continuous control with deep reinforcement learning [J]. Computer Science, 2016, 8(6): A187.

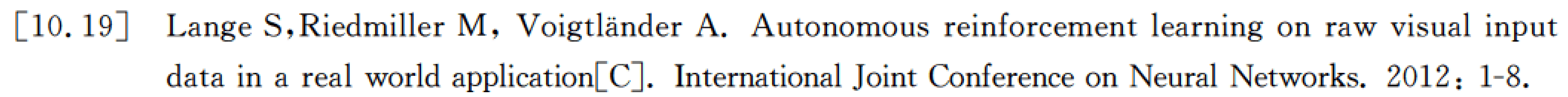
[10.15] 赵冬斌，邵坤，朱圆恒，等. 深度强化学习综述：兼论计算机围棋的发展[J]. 控制理论与应用，2016，33(6)：701-717.

[10.16] Lange S, Riedmiller M. Deep Auto-Encoder Neural Networks in Reinforcement Learning[J]. 2010: 1-8.

[10.17] Cuccu G, Luciw M, Schmidhuber J, et al. Intrinsically motivated neuroevolution for vision-based reinforcement learning[C]. IEEE International Conference on Development and Learning. IEEE Xplore, 2011: 1-7.

[10.18] Abtahi F, Fasel I. Deep belief nets as function approximators for reinforcement learning[C]. Lifelong Learning, Papers From the 2011 AAAI Workshop, San Francisco, California, Usa, August. DBLP, 2011: 2-7.



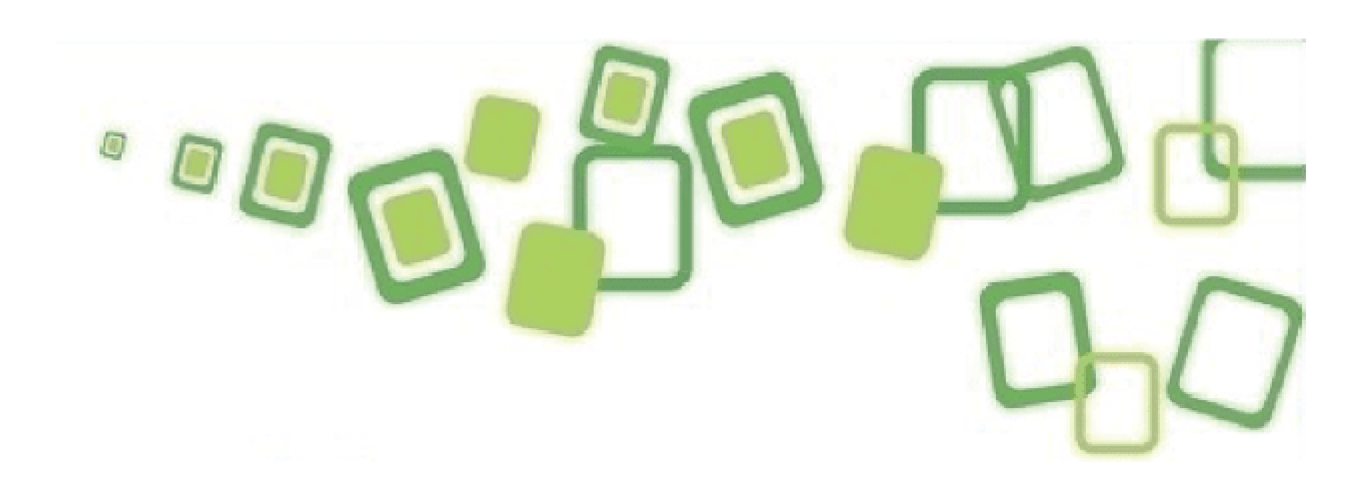


[10.19]　Lange S,Riedmiller M, Voigtländer A. Autonomous reinforcement learning on raw visual input data in a real world application[C]. International Joint Conference on Neural Networks. 2012: 1-8.

[10.20]　Arel I. Deep Reinforcement Learning as Foundation for Artificial General Intelligence[J]. 2012, 4: 89-102.

[10.21]　Mnih V, Kavukcuoglu K, Silver D, et al. Playing Atari with Deep Reinforcement Learning[J]. Computer Science, 2013.

[10.22]　Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529-533.

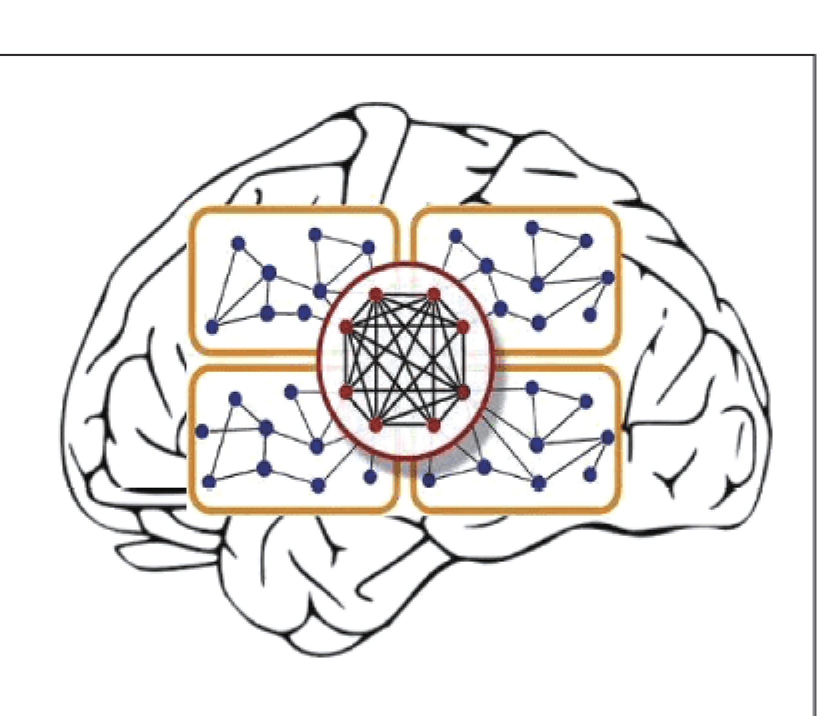
第11章



深度学习软件仿真平台及开发环境

CHAPTER 11

深度学习平台



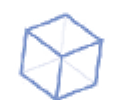
■ 深度神经网络模型复杂、训练数据多、计算量大

■ 超参数多、需反复多次试验

■ 多框架、大规模GPU集群

■ 并行迭代算法架构





11.1　Caffe 平台

11.1.1　Caffe 平台开发环境

Caffe 是深度学习框架之一，基于 C++语言编写，并且具有 licensed BSD，开放源码，提供了面向命令行，Matlab 和 Python 接口，是一个清晰，可读性强，快速的深度学习框架。作者是贾扬清。

Caffe 是通过 Layer 来完成所有的运算的。Caffe 定义一个网络模型，模型由多个 Layer 层组成，从数据层开始，loss 层结束。Caffe 是通过四维的 Blob 数据块来进行数据的存储和传递的，存储格式有 HDF5、LMDB 和 LevelDB 三种格式。目前使用最广泛的数据格式是 LMDB，HDF5 主要用于多标签分类。Caffe 之所以成为受欢迎的深度学习框架之一，主要有以下优势：

- Caffe 代码完全开源，速度快，支持 GPU 加速。
- Caffe 自带有一系列的网络模型，如 AlexNet、VGG、SSD 等，可以应用在多个研究领域。
- Caffe 代码设计具有模块化、可读性强的特点，便于新手学习。
- Caffe 具有 Python 和 Matlab 接口，灵活性强。
- Caffe 提供了一整套的数据处理流程，如数据预处理、训练、测试、精调。

Caffe 一般使用 Linux 系统，Windows 版的 Caffe 由于环境配置和使用时间较晚的问题，普及度不如 Linux 版本。本实验采用 Linux 版本。安装 Caffe 前，首先需要安装一些依赖工具，安装依赖库的时候可能会失败，这是下载所需的网站可能连接不上，请耐心等待。依赖库安装成功后，接着配置开发环境，为了达到最优的运行速度，需安装 CUDA 和 cuDNN，并配置相应的环境变量，前边所有的步骤安装成功后，就可以安装 Caffe 了，最后仍需要配置 Caffe 的环境变量并编译 Caffe，成功后，Caffe 就安装好了。具体的安装步骤可以参考 Caffe 官网。本实验的 Caffe 运行环境配置如下：

- 操作系统：Ubuntu 14.04。
- 显卡：Quadro K2200/PCIe/SSE2。
- CUDA 版本：cuda7.0。
- cuDNN 版本：cuDNN-v4.0。

11.1.2　AlexNet 神经网络学习

在图像识别方面，卷积神经网络(Convolutional Neural Networks，CNN)是一个非常热门的研究主题。最新的研究显示深度 CNN 模型在图像的特征提取和表示方面，取得了不错的进展。本节将介绍一个经典的深度学习模型——AlexNet。该网络结构是 2012 年

Alex 和 Hinton 参加 ILSVRC2012 比赛中所提出的网络模型，数据集是 ImageNet。这是 CNN 在 ImageNet 图像分类上的经典模型，之后的 VGG 模型和 CaffeNet 都是在 AlexNet 基础上改进的。该神经网络模型网络参数庞大，为了加快训练速度，使用了双 GPU 来处理卷积、池化运算，为了减少过拟合，全连接层后加了 Dropout 层，起到了很好的效果。在 ILSVRC2012 大赛中，top5 测试误差率为 15.3%，取得了第一名的好成绩。该网络是一个 8 层的卷积神经网络，由五个卷积层和三个全连接层、一个 softmax 损失层组成。因为该网络的结构深、参数多，所以和传统的 CNN 相比，可以得到更多的特征表达能力。具体的模型参数训练如下：

图像输入大小为 227×227×3，第一个卷积层滤波器的参数为 96×11×11×3，滤波器的个数为 96，大小为 11×11×3，步长为 4，卷积后的特征 map 的个数为 48。第一个卷积层的 ReLU，局部归一化、池化后的输出作为第二个卷积层的输入，第二个卷积层卷积核的参数设置为 256×5×5×48，第三个卷积层到第五个卷积层之间直接连接，这三个卷积层之间没有 ReLU 和池化。第五个卷积层进行 ReLU，池化后连接到第一个全连接层，共有 4096 个节点，第一个全连接层 ReLU，dropout 后输入到第二个全连接层，以此类推，连接到第三个全连接层，最后输入到 softmax 损失层，输出结果，每一层的具体的特征 map 数为 3-96-256-384-384-256-4090-4096-1000。

本模型采用 ReLU（非线性激活函数），其数学表达式如下：

$$\mathrm{ReLU}(x) = \max(0, x) \tag{11.1}$$

sigmoid 激活函数的数学表达式如下：

$$\mathrm{sigmoid}(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{11.2}$$

把输入的连续值归一化到 0～1 之间。

另外，tanh 的数学表达式如下：

$$\tanh(x) = \frac{\mathrm{e}^{x} - \mathrm{e}^{-x}}{\mathrm{e}^{x} + \mathrm{e}^{-x}} \tag{11.3}$$

ReLU 和 sigmoid、tanh 函数相比，数学公式简单，只有一个阈值 0，因此，计算速度快，没有梯度弥散的现象发生，而且在使用随机梯度下降法优化网络时，收敛比较快。

接下来，使用两个 GPU 来训练网络模型，一块简单的 GTX580 GPU 只有 3GB 的内存，120 万个训练样本，一个 GPU 显然是不够的。因此，使用两个 GPU 来传递网络参数。目前的 GPU 适合跨 GPU 并行化，它们能够不经过主机内存直接读取和写入彼此的内存。该并行化方案基本上将一半的神经元放在每个 GPU 上，并且 GPU 只在某些层中通信。例如，第二层的输出输入到第三层中，然而，第四层的神经元仅从位于同一 GPU 上的第三层中的神经元映射获取输入。使用两个 GPU，与在一个 GPU 上训练的每一个卷积层中一半的神经元相比，该方案将前 1 和前 5 的错误率分别减少了 1.7%和 1.2%。双 GPU 比单 GPU 网络需要更少的时间运行。

本模型使用了局部线性归一化，ReLU 激活函数具有不需要输入归一化来防止它们饱

和的特性。然而我们仍然发现以下局部归一化方案更具有普遍性。线性归一化公式如下：

$$b_{x,y}^{i}=\frac{a_{x,y}^{i}}{\left(k+\alpha\cdot\sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)}(a_{x,y}^{i})\right)^{2\beta}} \tag{11.4}$$

公式中 $a_{x,y}^{i}$ 表示在点 (x,y) 处，计算出的神经元的激活度，$b_{x,y}^{i}$ 表示归一化后的激活度，然后再通过 ReLU 激活函数，得到激活后的值。N 表示层中核的数量，常数 α,β,k,n 是超参数，一般设置为 $\alpha=10^{-4}$，$\beta=0.75$，$k=2$，$n=5$。这一策略可以降低 top1 和 top5 的误差率。

本模型采用了重叠 Pooling，CNN 中的池化层往往在卷积层之后，用来降低卷积层输出的特征，防止过拟合。池化通常有平均池化和最大池化两种。传统 CNN 模型，池化的相邻区域是不重叠的。本模型的池化选用的是池化区域重叠的最大池化。假设卷积后的特征图是由一个单元网格组成，池化半径为 n，如果相邻池化单元的中心位置的距离大于 n，就为传统 CNN 的不重叠池化，如果相邻池化单元的中心位置的距离小于 n，就是本模型中提到的重叠池化。和非重叠池化相比，重叠池化将 top1 和 top5 的误差率降低了 0.4%和 0.3%，而且重叠池化不容易过拟合。

该神经网络的参数大约有 6000 万，要学习这么多参数，很容易出现过拟合的情况，接下来介绍两种减少过拟合的方法：①数据扩充：减少数据最常用的方法就是增加数据量，可以在有限的训练样本的基础上，对数据进行镜像处理。例如我们把一张图切成 5 张小图，然后对其进行水平反射，就可以得到 10 张小图，数据扩充了一倍。或者，我们还可以对数据进行水平翻转、垂直翻转、平移、添加噪声等方法来增加数据量，减少过拟合。②Dropout：Dropout 就是以一定的概率，使一部分神经元激活，另一部分神经元抑制，概率属于超参数，由自己设定，通常情况下将概率设为 0.5，目的就是为了使网络结构稀疏化，降低网络的复杂度，它既不参与前向传播，也不参与反向传播。每一次层与层之间的信息传递都是不一样的神经元结构，参与信息传递的神经元的个数是固定的。通常 Dropout 层是在全连接层的后边，减少网络参数，防止过拟合。但是，Dropout 会增加收敛所需的迭代次数。

(1) 模型的优化，求解：

优化目标函数：Softmax withloss，计算公式如下：

$$\alpha_i=\frac{e^{z_i}}{\sum_{j=1}^{n}e^{z_j}} \tag{11.5}$$

$$L(\boldsymbol{W})=-\sum_{j=1}^{n}\log(\alpha_j) \tag{11.6}$$

(2) 求解方法：SGD(随机梯度下降法)，计算公式如下：

$$\boldsymbol{V}_t=\mu\cdot\boldsymbol{V}_t-\beta\cdot ?L(\boldsymbol{W}_t) \tag{11.7}$$

$$\boldsymbol{W}_{t+1}=\boldsymbol{W}_t+\boldsymbol{V}_t \tag{11.8}$$

其中 β 是负梯度学习率，μ 是上一次梯度值的权重。

11.1.3　AlexNet 神经网络应用于图像分类

本节将介绍 AlexNet 网络在自然图像分类中的应用。应用案例为花卉分类。该数据

集是在官网上下载的，一共有图片 1360 张，图片大小不同，有 17 类，每一类有 80 张图片。图片类别的具体介绍可参见其官网。官网链接如下：http://www.robots.ox.ac.uk/～vgg/data/flowers/17/index.html.

本实验从 1360 张图片中每类随机选取 80%作为训练样本，剩余的 20%作为测试样本。具体可以参见官网，部分样本如图 11.1 所示。

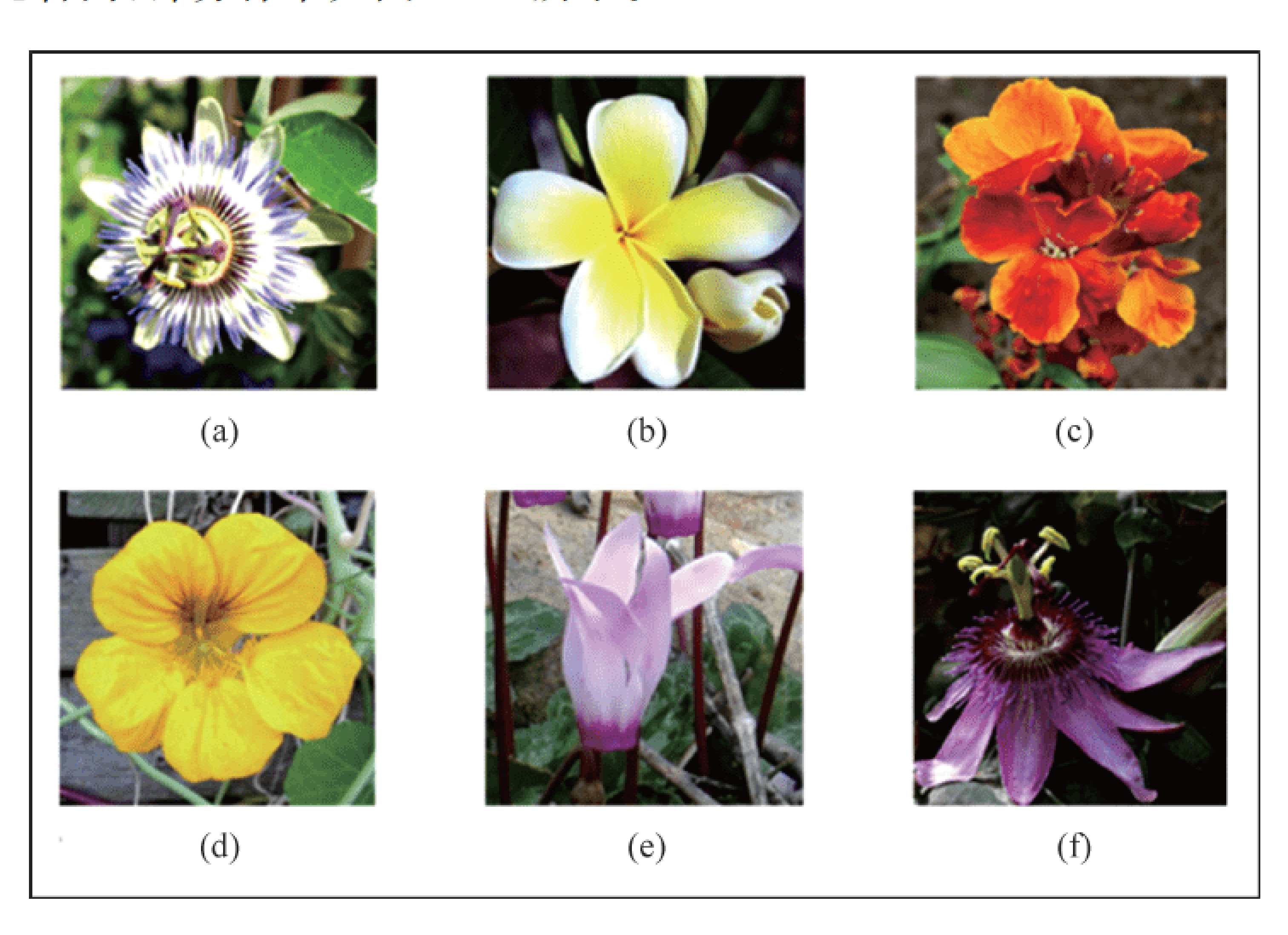


图 11.1　花卉数据集

网络部分超参数设置如下：#测试块大小为 50，训练块大小为 100。

- test_iter=5　#测试迭代 5 次，可以覆盖所有的测试集。
- test_interval：1000 #训练每迭代 1000 次，进行一次测试。
- base_lr：0.001 #学习策略：step(均匀分布策略)。
- lr_polocy：“step”。
- gamma：0.1。
- stepsize：5000。
- momentum：0.9 #网络的权重衰减和动量。
- weight_decay：0.0005。
- max_iter：20000 #最大迭代数。
- snapshot：5000 #每 5000 次迭代打印一次快照。
- snapshot_prefix：“/home/Downloads/Caffe/examples/flower/models/AlexNet”。
- solver_mode：GPU #Caffe 求解模式为 GPU。

实验结果展示如图 11.2 所示。

测试过程中的损失函数和准确率与迭代次数的关系如图 11.3 和图 11.4 所示。

```
I0108 16:10:04.467905 27603 solver.cpp:454] Snapshotting to binary proto file /h
ome/gaolili/Downloads/caffe/examples/flower/models/caffe_alexnet_train_iter_1000
0.caffemodel
I0108 16:10:08.217989 27603 sgd_solver.cpp:273] Snapshotting solver state to bin
ary proto file /home/gaolili/Downloads/caffe/examples/flower/models/caffe_alexne
t_train_iter_10000.solverstate
I0108 16:10:08.985585 27603 solver.cpp:337] Iteration 10000, Testing net (#0)
I0108 16:10:10.174505 27603 solver.cpp:404]     Test net output #0: accuracy = 0
.772
I0108 16:10:10.174651 27603 solver.cpp:404]     Test net output #1: loss = 1.811
34 (* 1 = 1.81134 loss)
I0108 16:10:11.471616 27603 solver.cpp:228] Iteration 10000, loss = 0.00106658
I0108 16:10:11.471730 27603 solver.cpp:244]     Train net output #0: loss = 0.00
106659 (* 1 = 0.00106659 loss)
I0108 16:10:11.471765 27603 sgd_solver.cpp:106] Iteration 10000, lr = 1e-05
I0108 16:12:53.242400 27603 solver.cpp:228] Iteration 10100, loss = 0.000390353
I0108 16:12:53.243039 27603 solver.cpp:244]     Train net output #0: loss = 0.00
0390365 (* 1 = 0.000390365 loss)
```

图 11.2　实验结果图

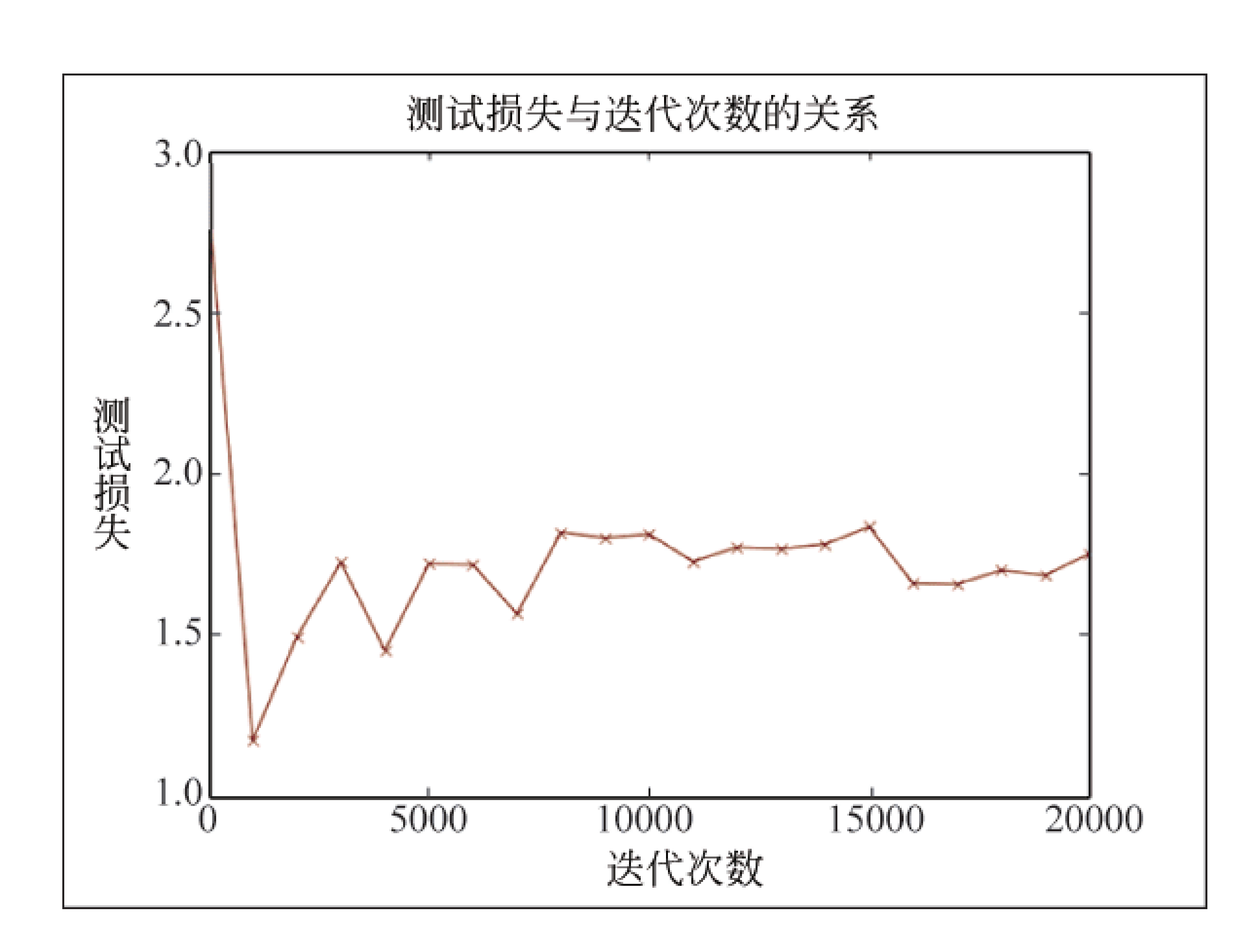


图 11.3　损失函数 loss 曲线图

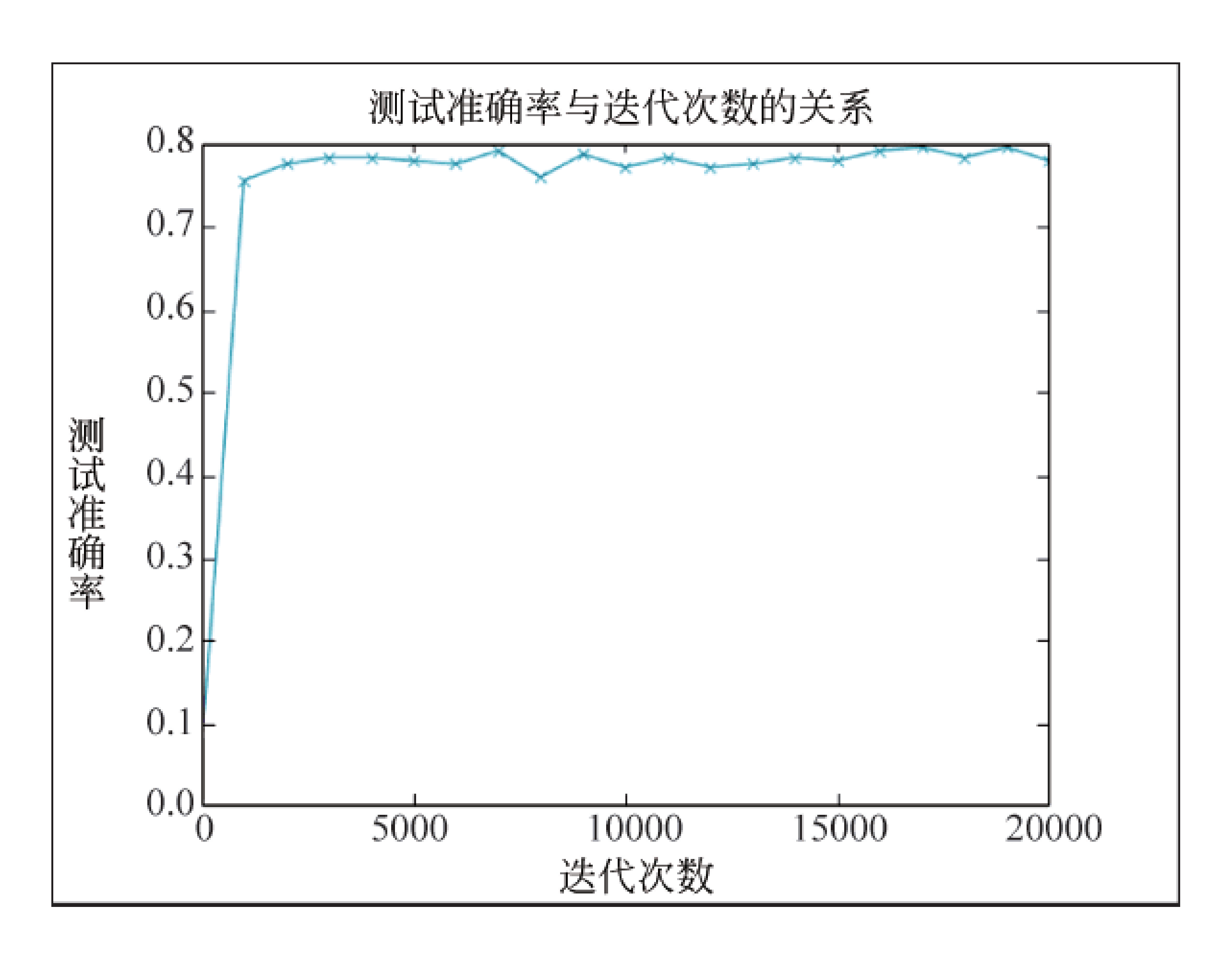
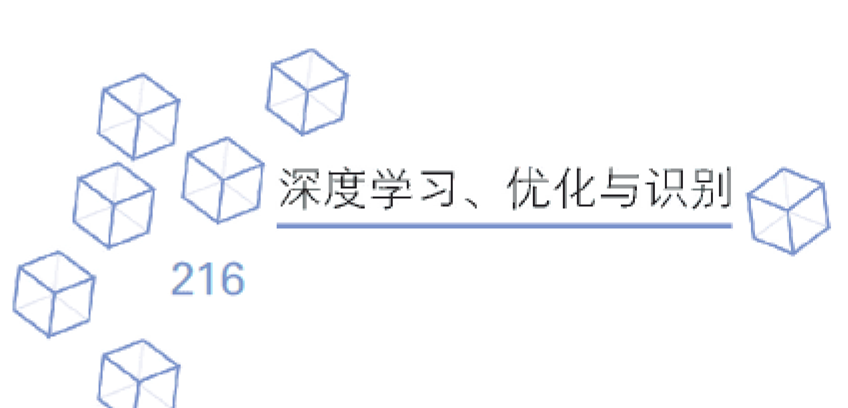


图 11.4　准确率 accuracy 曲线图



11.2 TensorFlow 平台

11.2.1 TensorFlow 平台开发环境

TensorFlow 是 Google 公司在 2015 年底发布的开源人工智能系统。该系统此前一直是 Google 公司的内部机器学习系统，该系统的开源极大地方便了广大机器学习研究工作者的科研工作。TensorFlow 架构灵活，很多平台上都可以使用，支持一个或多个 CPU，或者一个或多个 GPU、服务器、移动设备等。而且目前 v0.8 版本已经支持分布式计算(即云计算)。

TensorFlow 是一个采用数据流图，由“节点”和“线”组成来进行数学计算的开源软件库。一个图中的每一个节点表示一个数学操作，每一条线表示节点之间的输入、输出关系。节点之间流通的是数据，这些数据用“张量”(tensor)这种数据结构表示，用线来进行运输。而至于“张量”，我们则可以想象成 Python 语言里的一个 N 维的数组或者列表。每一个节点即操作，首先会获得零个或多个张量，然后执行操作，再产生零个或多个张量。一个 TensorFlow 图描述了数学计算任务的过程。而在执行计算的时候，需要将图在“会话”(session)中启动。在 TensorFlow 上进行开发编程的常规流程就是：首先创建一个图，然后在“会话”中加载图。

TensorFlow 还能可视化学习。在 TensorFlow 发布包中有一套叫做 TensorBoard 的可视化工具，它可以用来可视化 TensorFlow 计算任务中的数据流图(Graph)、定量指标图和附加数据。图 11.5 是 11.2.3 节中 Mnist 手写体数据集实验中的可视化结果。TensorFlow 有很多优点：

- 高度的灵活性，TensorFlow 并不是一个严格的深度学习框架范围内的开发系统，任何可以转化为数据流图形式的计算都可以被使用。TensorFlow 也是一个很底层的框架，可以根据需要在 TensorFlow 上开发上层的库。
- 可移植性好，TensorFlow 可以在 CPU 和 GPU 上运行，不管是台式机、笔记本，还是移动设备，不管是将模型作为云端服务，还是将其运行在 Docker 容器里，TensorFlow 都能满足需求。
- 能自动求微分，因为机器学习中很多基于梯度的算法，而 TensorFlow 能自动为用户计算相关的导数。
- 支持多种开发语言，现在主要支持 Python 和 C++，并且有合理易用的界面。
- 最优化性能，TensorFlow 对队列、线程、异步操作等有最佳的处理，不管用户有多优秀的硬件，TensorFlow 都能将硬件的性能发挥出来。
- 科研与产品都支持的系统。

TensorFlow(其界面如图 11.5 所示)一般都是在 UNIX 内核环境下运行，如 Linux 操作系统。虽然利用 Docker 也可以将其运用于 Windows 上，但是会出现很多问题。目前社



图 11.5 TensorBoard 界面

区主要推荐的 Linux 版本 Ubuntu 14.04，因为 TensorFlow 主要还是基于 Python 2 的。而更高版本的 Ubuntu 16 默认都是 Python 3。本章实验都是在 Ubuntu 14.04 版本的操作系统下进行的。TensorFlow 所支持的开发语言主要有 Python 和 C++，但使用广泛和支持最好的还是 Python 语言，因此，本章实验的开发语言都是 Python 语言。Python 是一种高级语言，用其编写代码十分简洁，但由于相比 C++这种底层语言，Python 简洁易用，所以速度会比较慢。TensorFlow 的安装方法在其官网上也有列出，目前有三种安装方法。本章实验 TensorFlow 安装环境的 CUDA 版本为 7.5，cuDNN 版本为 7.0。

11.2.2 深度卷积生成式对抗网 DCGAN

本节主要介绍 2015 年 Radford A 等人在 Computer Science 上发表的论文 *Unsupervised representation learning with deep convolutional generative adversarial networks* 中提出的深度卷积生成式对抗网(Deep Convolution GAN，文中缩写为 DCGAN，现在通用缩写 DAN-Convolutions)，在 TensorFlow 平台上有相关实现代码。DAN 结合有监督学习的 CNN 和无监督学习的 GAN，能够进行无监督表征学习，训练好的生成器和判别器的隐含层都可以对图像进行特征表示。DAN 的结构是在原始 GAN 的基础上，将生成器和判别器的隐含层全部用卷积层实现。虽然 GAN 本身训练不需要特定的启发式损失函数，优化过程是一个“二元极大极小博弈”问题，但是 GAN 本身训练十分不稳定。本文作者根据自己在 CNN 领域的工程经验，提出和评估了一系列约束使得网络在训练中稳定。其模型的方法和核心主要有：

- 用生成器的带步长卷积(strided convolutions)替换所有池化层。
- 用判别器的微步卷积(fractional strided convolutions)替换所有池化层。

- 在生成器和判别器上都使用批标准化(batch normalization),这个策略能有效地解决初始化不当会引起训练崩溃的问题,但如果将批标准化应用于所有层又会引起模型的不稳定,所以采取的措施为在生成器的输出层和判别器的输入层不使用批标准化。
- 删除深度网络中的全连接层,论文中提到,原始 CNN 中一般使用的全局池化(global pooling),这样虽然可以增加模型的稳定性,但是网络的收敛速度会降低。
- 生成器中输出层用 Tanh 激活函数,其他所有层用 ReLU 激活函数。
- 判别器中所有层的激活函数都用 Leaky ReLU。

具体网络模型如图 11.6 所示。对于生成器 G,输入为 100 维的均匀噪声,第一层为全连接层,将 100 维的向量投影成 4×4 大小的 feature map,通道数为 512。然后依次用四层步长为 5×5 的带步长卷积,这样使得每次卷积后图像尺寸加倍,通道数减半。最后转换为 64×64 大小的 RGB 三通道图片,这些图片就是生成的假样本。对于判别器 D,输入为真实的样本和生成器生成的伪样本,均为 64×64 大小的 RGB 三通道图片。判别器与生成器的各层的图像尺寸和通道数保持一致。判别器的前四层图像尺寸依次减半,通道数加倍,生成高级特征表示。最后一层为一个 logistics 回归二分类器,输出为一个标量,即对样本真实性的评分,表示是否为真实样本。

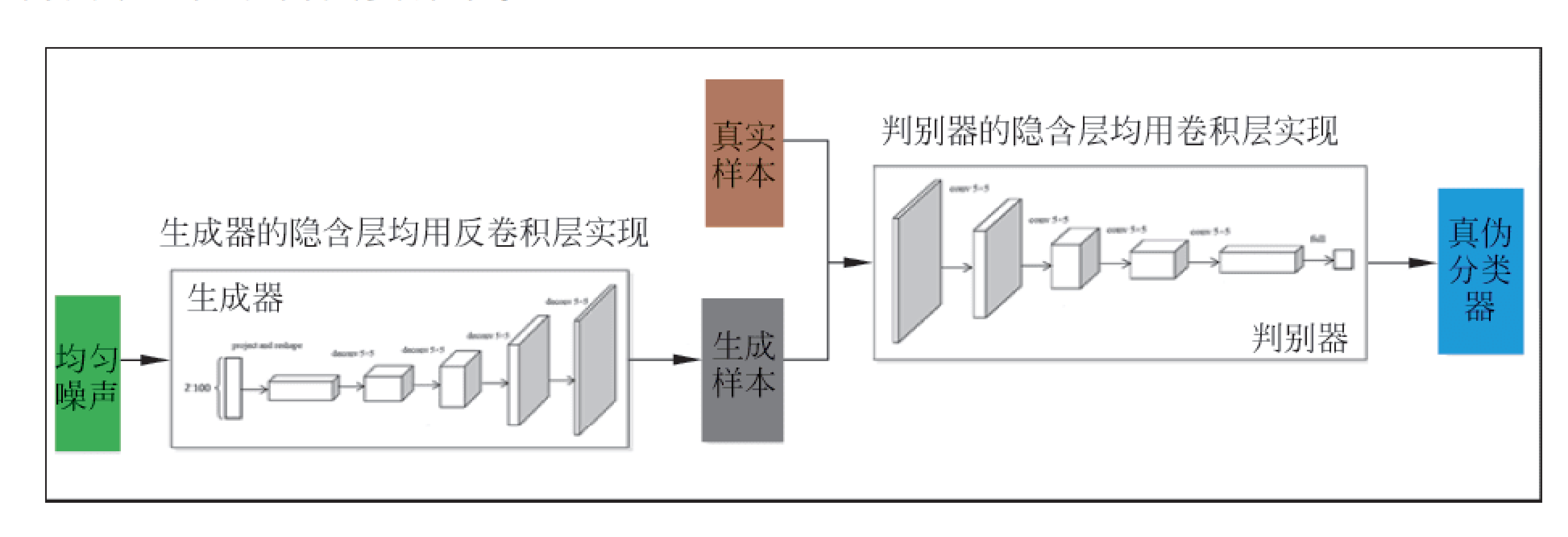


图 11.6 生成式对抗网络 GAN 网络结构

训练的一些超参设置:训练采用 mini-batch 进行训练,训练的 batch size 为 64;以往的 GAN 都采用 momentum 优化器加速训练,DAN 采用 Adam 优化器进行学习训练,且学习率为 0.0002;所有的参数初始化都是从正态分布得到的,正态分布设置的参数均值为 0,方差为 0.02;设置 Leaky ReLU 的斜率为 0.2;将 momentum 参数 beta 从 0.9 降为 0.5,有助于训练稳定,防止震荡。

11.2.3 DAN 应用于样本扩充

根据上一节中介绍的网络模型,可以进行样本扩充,而且 DAN 具有无监督表征能力,这样可以将其用于特征提取从而进行图像分类。在下一章中有关于利用 DAN 的无监督表征,结合有监督学习进行极化 SAR 分类的章节。本章实验采用 Mnist 手写体数据集和 flower(102 类)数据集进行样本扩充,模拟样本数据的内在分布特性。因为手写体大小为

28×28，与上一节所述的通用大小不同，这里对网络模型做了一些调整，并且将类标同时送入网络中，这里就不详细介绍了，下面是生成的手写体样本。图 11.7 是第 1 代生成的手写体样本，可以看到此时比较模糊，并不能很完整地辨识出具体的数字，但已经有了手写体的雏形。而且我们发现第一代生成的数字质量和多样性都比较差。这里没有显示最开始 100 维噪声的图片，其实就是一些杂乱的黑白点。图 11.8 是第 10 代生成的手写体样本。此时，相比第 1 代，网络已经渐渐学习到了数据集的分布特性，能够生成相对清晰的手写体数字，而且与原数据很相像。但是笔画仍然断断续续，数字体外还是有多余的白点。图 11.9 是第 20 代网络生成的手写体样本，显而易见，此时的数字相较于以前的样本已经十分清晰，我们可以用肉眼识别出具体的数字，只是个别的数字会有笔画不清晰的情况。图 11.10 是第 25 代生成的手写体样本，可以看出生成的样本足以“以假乱真”。

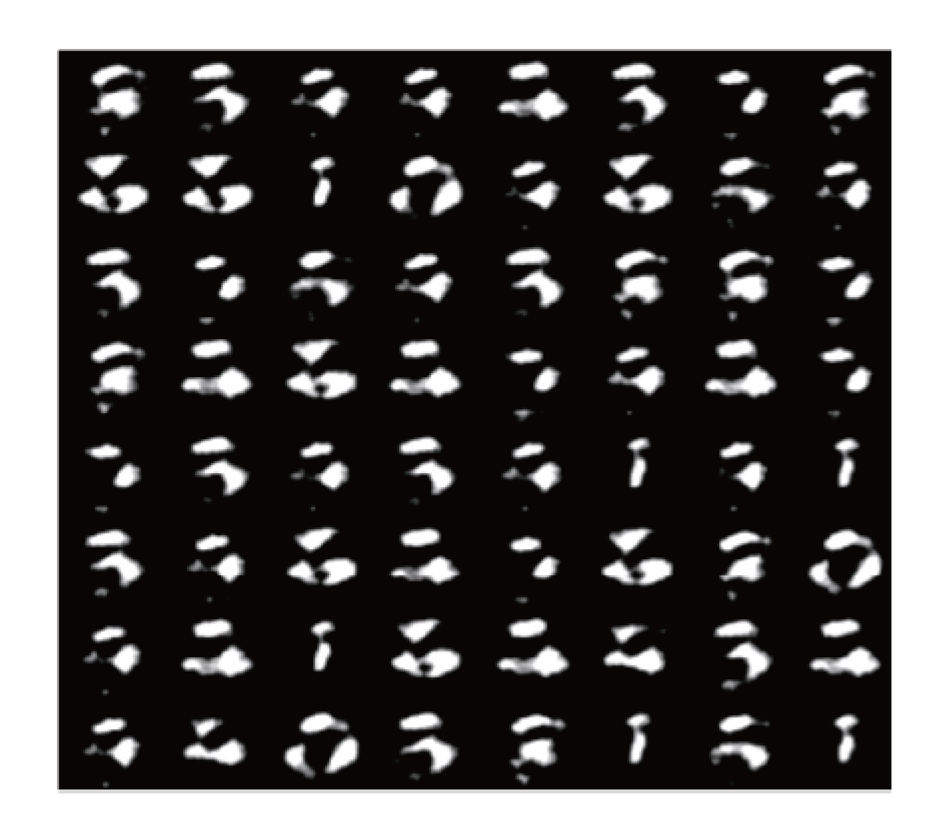


图 11.7　第 1 代生成样本



图 11.8　第 10 代生成样本



图 11.9　第 20 代生成样本



图 11.10　第 25 代生成样本

另外，我们用 TensorBoard 可以再看一下两个模型在训练过程中的 Loss。图 11.11 是判别器的 loss，图 11.12 是生成器的 loss，图中横坐标均为训练代数。由图可以看出两个模型都存在一定的抖动，特别是刚开始训练时抖动比较厉害，这也算是两个模型在对抗过程中



此消彼长。在以后不断训练的过程中,两个模型都趋于稳定,只是稍微有些起伏。这也显示了这个模型相比一般的GAN的稳定性。

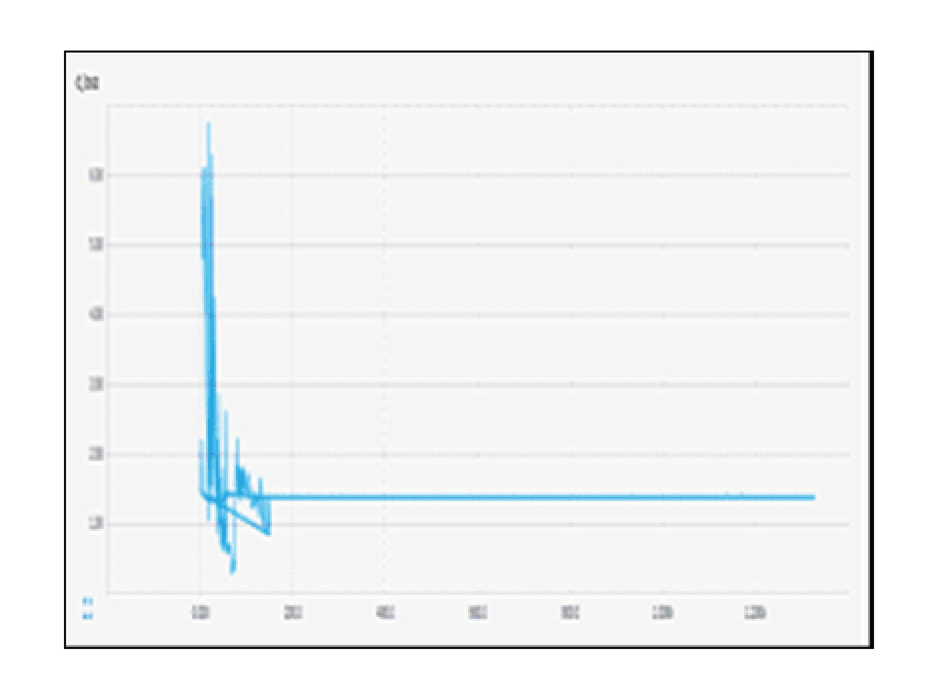


图 11.11　判别器的 loss

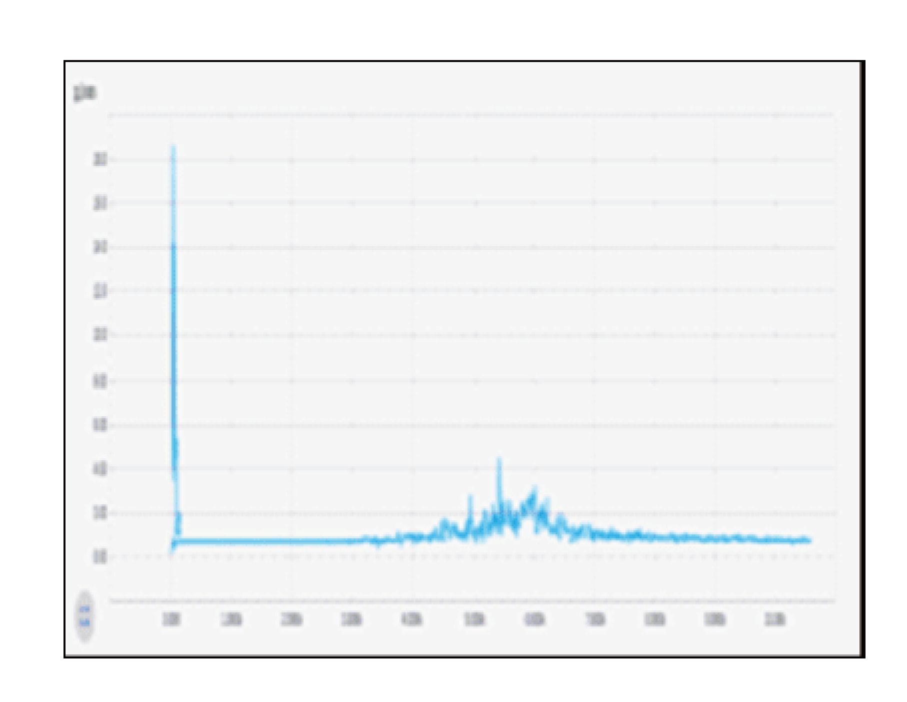


图 11.12　生成器的 loss

我们还用 flower102 数据集对网络进行了测试,观察生成的样本。这个花朵数据集是 Nilsback、M-E 和 Andrew Zisserman 收集的在英国盛开的花朵图片。数据共分为 102 类,每个类别由 40 到 258 个图片组成,本章实验下载使用的花朵数据集中共 8189 张花朵图片。该数据集的特点在于规模、拍摄角度和光线差别都比较大。在数据集中有差别较大的类别,也有比较相似的类别。我们可以用 Isomap(等距映射)算法根据它的形状 shape(图 11.13)和颜色 color(图 11.14)可视化这个数据集。

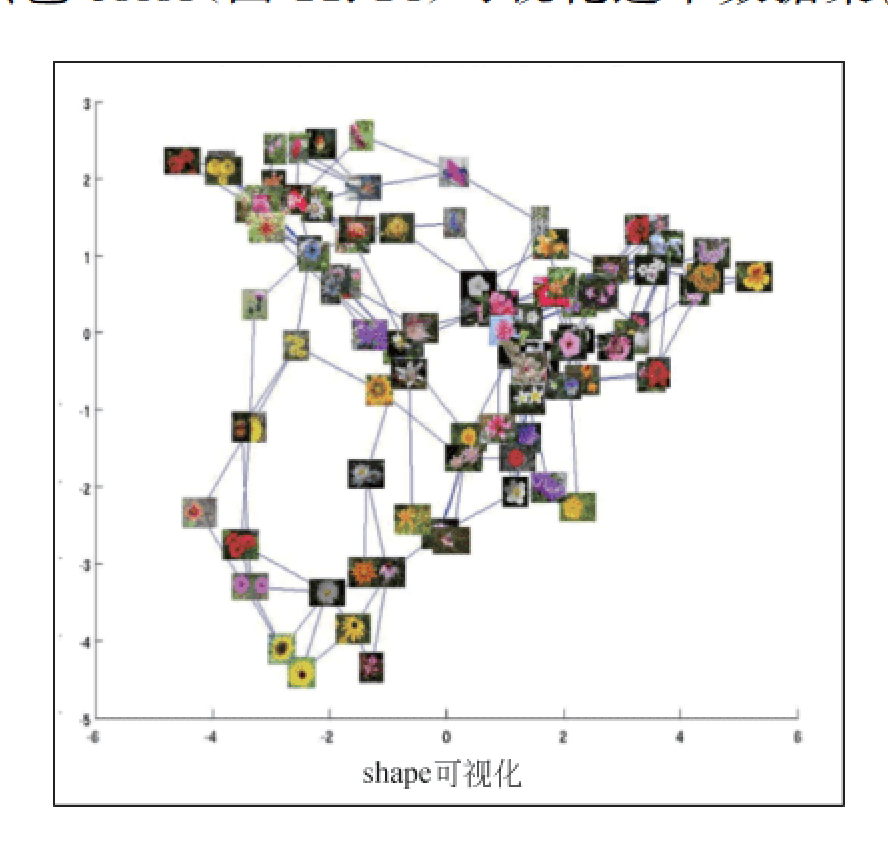


图 11.13　shape 可视化

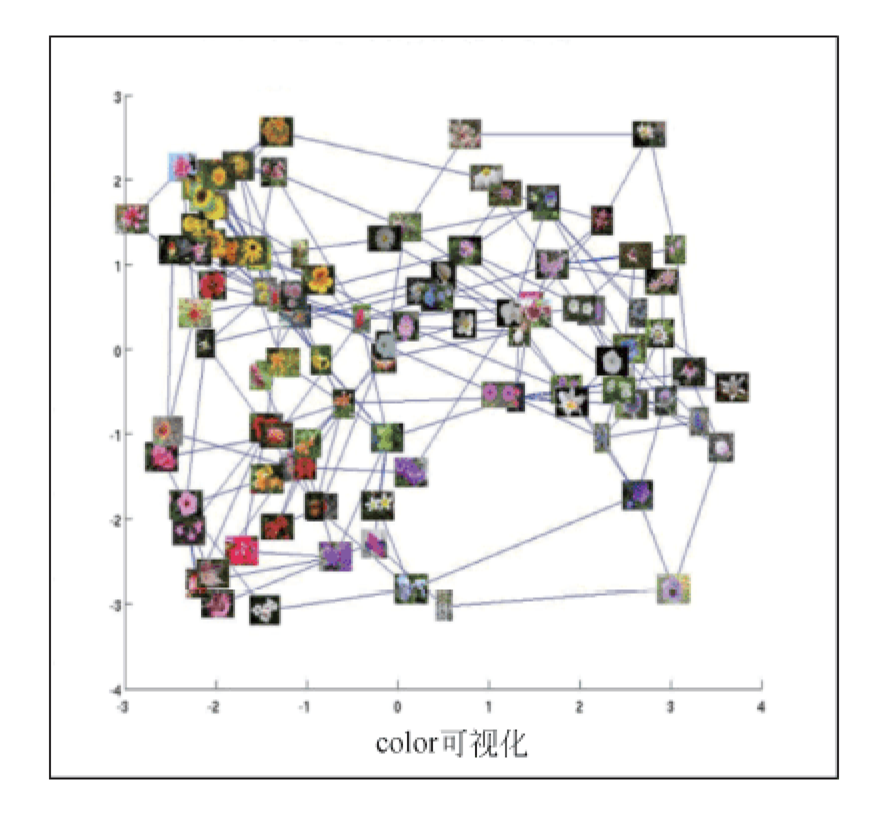


图 11.14　color 可视化

因为数据集中图片大小不规则,所以我们在将数据送入网络之前,将所有图片都处理成 64×64 大小。下面是实验结果,每张结果图显示了 8×8 个 64×64 的生成样本。图 11.15 是第一代生成的花朵图片,由于网络开始是随机初始化权值,此时的生成器还不会生成图片,也可以通俗地说成是“傻”的,因此会生成无意义的图片,作为判别器的训练集假的部分。图 11.16 是第 20 代生成的花朵样本。由图可见,生成的图片已经隐约有了花朵的色彩和大致形状。由于判别器可以把真图和假图识别出来,我们再训练生成器,使得其输出能欺骗判

别器，这样生成器能窥探到判别器的损失函数，生成越来越相像的样本。图 11.17 是第 60 代生成的花朵图片，比之前的轮廓更清晰。图 11.18 是第 80 代的生成样本，此时花朵已经很逼真了。

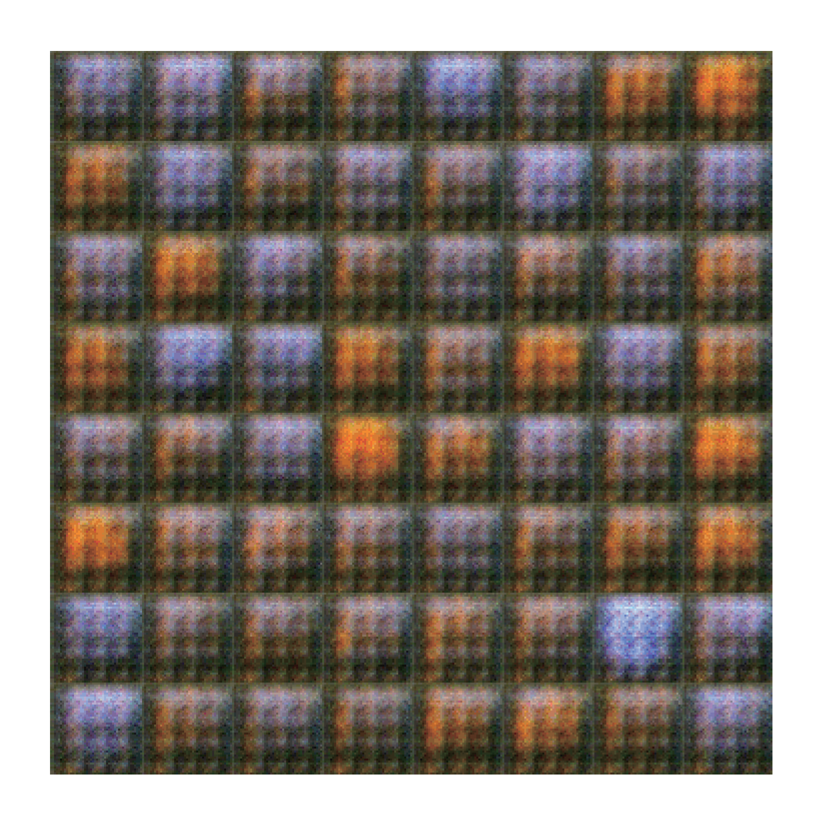


图 11.15　第 1 代生成样本

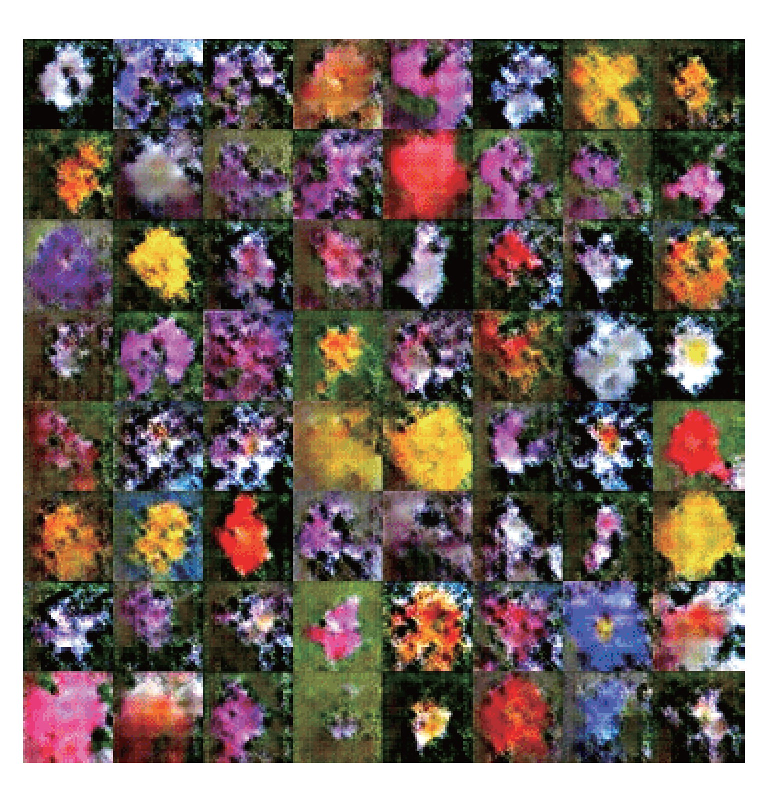


图 11.16　第 20 代生成样本



图 11.17　第 60 代生成样本



图 11.18　第 80 代生成样本

11.3　MXNet 平台

11.3.1　MXNet 平台开发环境

MXNet 是由 dmlc/cxxnet、dmlc/minerva 和 Purine2 的作者发起的，是一个效率高、灵活性强的深度学习框架，兼具 Minerva 的动态执行，cxxnet 的静态优化和 Purine2 的符号计算的思想，支持基于 Python 的 parameter server 接口，保证代码可以很快向分布式进行迁移。MXNet 所有模块的设计简洁清晰，可以方便地被使用。C 接口和静态、动态库设计使



得该平台扩展新语言变得简单可行。目前现有的深度学习平台大部分都是采用命令式编程和符号式编程中的一种，与它们不同的是，MXNet 尝试将两种编程模式进行无缝结合，允许开发者使用混合符号编程和命令式编程。在符号式编程中，MXNet 支持符号表达，在命令式编程中，MXNet 则支持张量运算，混合式编程最大限度地提高了效率和生产力，用户可以通过使用两种编程方式来实现自己的思想。MXNet 的核心是一个动态的依赖调度，能够实现符号和命令操作的自动并行。它的图形优化层加快了符号的执行速度，提高了内存的使用效率。MXNet 的库具有便携、轻量的特点，此外还可以方便地扩展到多个 GPU 和多台机器。

MXNet 可进行多设备间的数据交互，该功能通过 KVStore 实现，它提供一个分布式的 key-value 存储来进行数据交换，主要包括两个函数：push 函数，实现从一个设备将 key-value 放进存储的功能；pull 函数：实现从存储中将一个 key 中的值 pull 出来的功能。当用户想在 Linux 或 Ubuntu 上运行 Python/R 时，可以使用 Git Bash 脚本快速安装 MXNet 库以及其他依赖。MXNet 还可以在 Docker 或者云上(例如 AWS)运行，除此之外，MXNet 还具有如下特点：

- 支持平台：Ubuntu/Debian，OS X，Windows，AWS，Android，iOS，JavaScript。
- 编写语言：C++，Python，Julia，Matlab，R，Scala。
- 支持 CUDA。
- 支持云计算：所有数据模型可以从 S3/HDFS/Azure 上直接加载训练等。

MXNet 的系统架构见图 11.19。从上到下分别为各种主语言的嵌入，编程的接口(矩阵运算、符号表达式、分布式通信)，两种编程模式的统一实现，以及各硬件的支持。

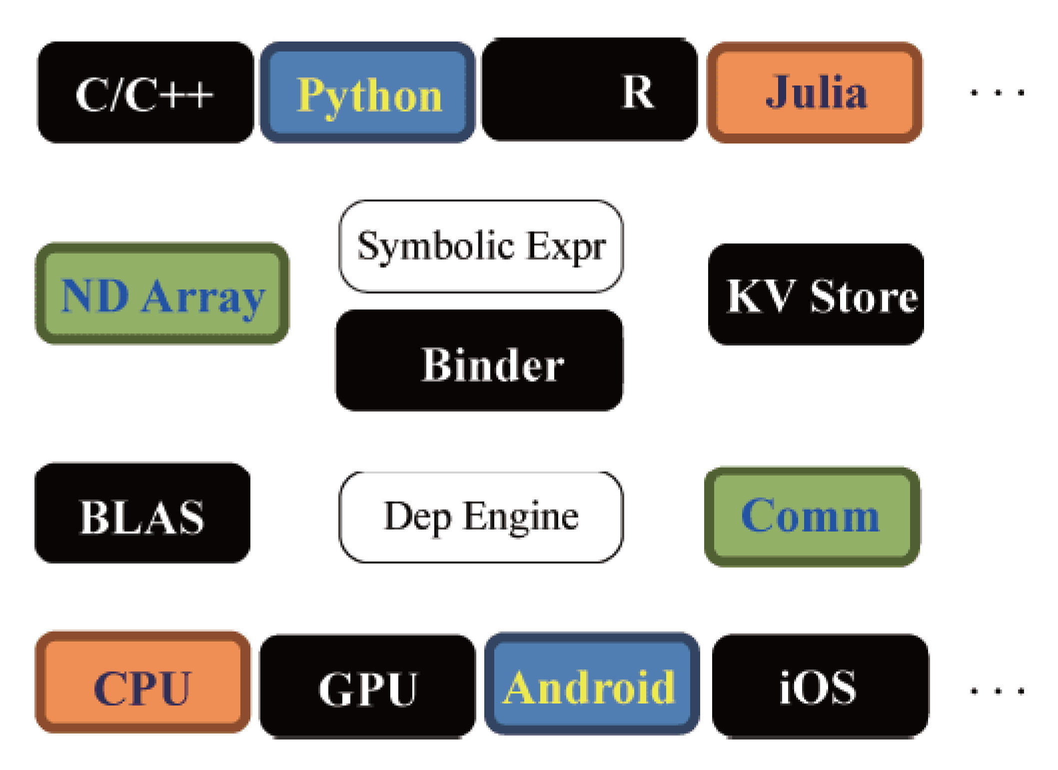


图 11.19　MXNet 的系统架构

MXNet 的开源社区为用户提供了有力的支持。

- 广泛的模型支持——训练和部署了最新的深度卷积神经网络(CNN)和短期记忆模型等。
- 广泛的参考示例库——建立示例教程(包括代码)，如图像分类、语言建模、神经艺术、语音识别等。
- 开放和协作的社区——社区支持和贡献者很多都来自世界顶级大学和商业合作者。

11.3.2　VGG-NET 深度神经网络学习

VGG-NET 是由 Andrew Zisserman 教授的团队设计的，它是卷积神经网络的一种，用来解决大规模图像分类问题，曾在 2014 年的 ILSVRC localization 和 classification 两个问题上分别取得了第一名和第二名的好成绩。VGG-NET 在原理上与普通的 CNN 并无不同，其最大的特点有两个：一是局部感受野小，全部采用 3×3 的滤波器；二是网络结构深，通常有 16～24 层。常见的 VGG-NET 网络结构见表 11.1。由表可知，6 种 VGG-NET 网络的输入均为 224×224 大小的 RGB 图像，有 5 个最大池化层，三个全连接层（前两层各有 4096 个神经元，最后一层有 1000 个神经元），一个 soft-max 层，不同网络的卷积部分级联的卷积层个数不同，卷积层从 8 到 16 不等，卷积层滤波器的个数从 64 开始，最后达到 512。由表 11.1 可知，VGG-NET 的 C 中使用了 1×1 的卷积滤波器，可以看作输入的线性变换，除此之外，所有滤波器的大小都设置为 3×3，步长固定为 1，最大池化为 2×2，步长为 2。所有隐层的激活函数均采用非线性激活函数 ReLU。VGG-NET 中除了 A-LRN 其他网络均未进行局部响应归一化（LRN），这是因为在 VGG-NET 中使用 LRN 不仅不会提高分类准确率，反而会增加内存消耗和计算时间。

整个 VGG-NET 中滤波器的大小都为 3×3，使用两个具有 3×3 的滤波器的卷积层级联的效果等同于使用一个具有 5×5 的滤波器的卷积层的效果。使用三个具有 3×3 的滤波器的卷积层级联的效果等同于使用一个具有 7×7 的滤波器的卷积层的效果，相较于一个 7×7 的滤波器，三个 3×3 的滤波器对应三个非线性激活函数 ReLU，这使得判决函数更具有判决能力，同时还可以减少参数个数，假设级联的三个卷积层的滤波器大小都为 3×3，输入输出的通道数都为 C，则参数个数为 $3\times(3\times3\times C\times C)=27\times C\times C$，一个卷积层的滤波器大小为 7×7，则参数个数为 $7\times7\times C\times C=49\times C\times C$，显然具有 3×3 的滤波器的三层卷积层级联的参数更少。作为 CNN 的一种，VGG-NET 由卷积层和全连接层组成。网络中的卷积层描述了输入的局部特征，全连接层则描述了输入的全局特征，尤其是 FC3 层可以很好地描述全局特征，所以该层常被当作输入数据的新的特征，用来实现图像分类。网络权值的初始化非常重要，较差的初始值会使深度网络中的梯度不稳定，从而导致网络无法继续学习，为了解决该问题，VGG-NET 使用了预训练的方法，首先构建拥有较少网络层数的 A 来进行训练，当训练更深的网络时，前四个卷积层和最后的三个全连接层使用训练好的 A 的参数进行初始化，其余参数随机初始化。表 11.2 给出了 VGG-NET 的参数个数，这些参数是如何计算得到的？以 A 为例，输入为 224×224 的 RGB 图像，第一个卷积层有 64 个大小为 3×3 的滤波器，因此第一个卷积层参数个数为(3×3×3)×64＝1728，同理，各层参数计算见表 11.3。虽然 VGG-NET 的网络层数很多，但是感受野很小，因此整个网络的参数个数相对来说并不多。表 11.1 中的 D、E 是我们常见的 VGG-16 和 VGG-19 模型。

此处，我们重点介绍 MXNet 平台的 example 中给出的 VGG-NET 即 VGG-A，其网络结构见图 11.20。VGG-NET 最初提出时是用于 ILSVRC-2012 数据集分类的，因此 VGG-A 中的 soft-max 分类器有 1000 个神经元。VGG-A 中除去输入和输出共有 16 个网络层，其中有 8 个卷积层、5 个池化层和 3 个全连接层。

表 11.1　VGG-NET 网络结构

<table>
<tr><th>A</th><th>A-LRN</th><th>B</th><th>C</th><th>D</th><th>E</th></tr>
<tr><td>16 layers</td><td>16 layers</td><td>18 layers</td><td>21 layers</td><td>21 layers</td><td>24 layers</td></tr>
<tr><td>conv3-64</td><td>conv3-64
LRN</td><td>conv3-64
conv3-64</td><td>conv3-64
conv3-64</td><td>conv3-64
conv3-64</td><td>conv3-64
conv3-64</td></tr>
<tr><td colspan="6">max-pool</td></tr>
<tr><td>conv3-128</td><td>conv3-128</td><td>conv3-128
conv3-128</td><td>conv3-128
conv3-128</td><td>conv3-128
conv3-128</td><td>conv3-128
conv3-128</td></tr>
<tr><td colspan="6">max-pool</td></tr>
<tr><td>conv3-512
conv3-512</td><td>conv3-512
conv3-512</td><td>conv3-256
conv3-256</td><td>conv3-256
conv3-256
conv1-256</td><td>conv3-256
conv3-256
conv3-256</td><td>conv3-256
conv3-256
conv3-256
conv3-256</td></tr>
<tr><td colspan="6">max-pool</td></tr>
<tr><td>conv3-512
conv3-512</td><td>conv3-512
conv3-512</td><td>conv3-512
conv3-512</td><td>conv3-512
conv3-512
conv1-512</td><td>conv3-512
conv3-512
conv3-512</td><td>conv3-512
conv3-512
conv3-512
conv3-512</td></tr>
<tr><td colspan="6">max-pool</td></tr>
<tr><td>conv3-512
conv3-512</td><td>conv3-512
conv3-512</td><td>conv3-512
conv3-512</td><td>conv3-512
conv3-512
conv1-512</td><td>conv3-512
conv3-512
conv3-512</td><td>conv3-512
conv3-512
conv3-512
conv3-512</td></tr>
<tr><td colspan="6">max-pool</td></tr>
<tr><td colspan="6">FC 4096</td></tr>
<tr><td colspan="6">FC 4096</td></tr>
<tr><td colspan="6">FC 1000</td></tr>
<tr><td colspan="6">softmax</td></tr>
</table>

表 11.2　VGG-NET 网络参数

网　　络	A	A-LRN	B	C	D	E
参数数量/百万	133	133	133	134	138	144

表 11.3　VGG-A 参数数量计算

网　络　层		参　数　个　数
conv1	224 * 224 * 64	(3 * 3 * 3) * 64=1728
max-pool	112 * 112 * 64	0
conv2	112 * 112 * 128	(3 * 3 * 64) * 128=73 728
max-pool	56 * 56 * 128	0
conv3	56 * 56 * 256	(3 * 3 * 128) * 256=294 912
	56 * 56 * 256	(3 * 3 * 256) * 256=589 824
max-pool	28 * 28 * 256	0

续表

网　络　层		参 数 个 数
conv4	28 * 28 * 512	(3 * 3 * 256) * 512=1 179 648
	28 * 28 * 512	(3 * 3 * 512) * 512=2 359 296
max-pool	14 * 14 * 512	0
conv5	14 * 14 * 512	(3 * 3 * 512) * 512=2 359 296
	14 * 14 * 512	(3 * 3 * 512) * 512=2 359 296
max-pool	7 * 7 * 512	0
FC1	1 * 1 * 4096	(7 * 7 * 512) * 4096=102 760 448
FC2	1 * 1 * 4096	4096 * 4096=16 777 216
FC3	1 * 1 * 1000	4096 * 1000=4 096 000
softmax	1 * 1 * 1000	1000 * 1000=1 000 000

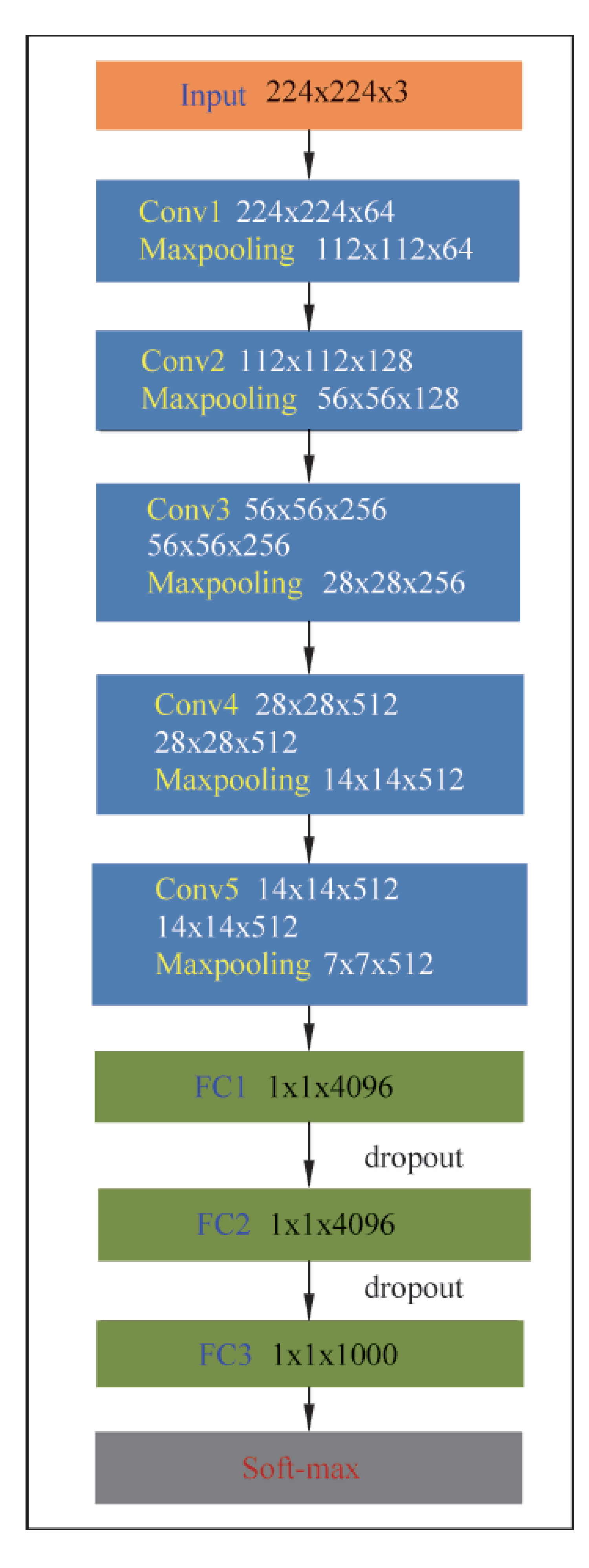
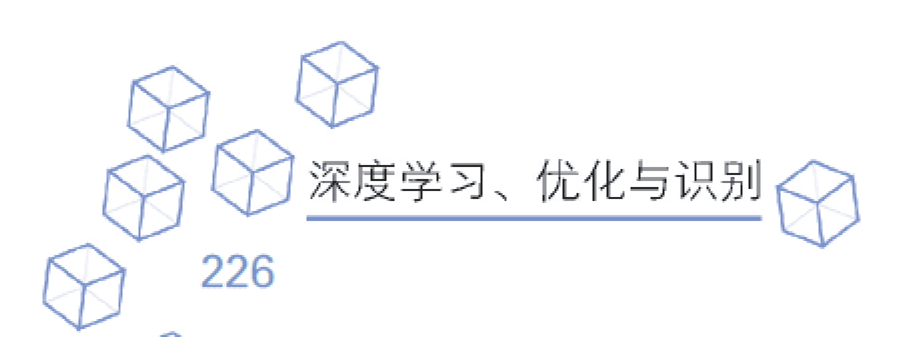


图 11.20　VGG-A 网络结构图



11.3.3 图像分类应用任务

使用 VGG-A 对 Flower 数据集进行分类。Flower 数据集共有 102 个类别，8189 个样本，每类包含 40 到 258 个不等的样本数。每张图片的尺寸不同，需对其进行归一化，由于 VGG-A 的输入是 224×224 的 RGB 图像，全部归一化为 224×224。随机选取其中 1000 个样本组成测试集，其余 7189 个样本组成训练集。

symbol_vgg. py 文件实现 VGG-A 网格结构的构建，由于类别数为 102，num_classes＝102。train_model. py 文件实现的功能有多设备间的数据交互，加载模型、保存模型等。train_flower. py 文件对相关参数进行设定，加载数据，设置迭代器，对模型进行训练等。由于数据是以 numpy 类型存储的，因此使用了 NDArrayIter 迭代器。若数据是以 MXNet 特有的 rec 格式存储的，则可使用 ImageRecoderIter 迭代器。迭代器及主要参数设定如下：

```
def get_iterator(args,kv)
        train = mx.io.NDArryIter(
data = train_data,
label = train_label,
batch_size = args.batch_size,
shuffle = True,
last_batch_handle = 'pad'
)
    val = mx.io.NDArryIter(
        data = test_data,
    label = test_label,
    batch_size = args.batch_size,
    shuffle = True,
    last_batch_handle = 'pad'
    )
    return train,val
lr = 0.005
num - epoch = 120
batch - size = 100
```

默认不进行多设备间的数据交互，运行 train_imagenet. py 文件即可开始对 VGG-A 进行训练，如图 11.21 所示，训练结果如图 11.22 所示。训练时使用了一个 GPU，训练一个 epoch 用时 3 分钟左右，每训练一个 epoch 进行一次测试。最终的训练准确率为 89.2%，top5 训练准确率为 93.9%，测试准确率为 56.9%，top5 测试准确率为 80.7%。VGG-NET 的网络很深，训练集样本数过少，导致模型过拟合，读者可使用更大的数据集来进行实验，相信会获得更好的分类结果。

```
dlg@IPIU3:~$ cd /home/dlg/Documents/VGG
dlg@IPIU3:~/Documents/VGG$ python train_flower.py
2017-01-11 15:08:04,739 Node[0] start with arguments Namespace(batch_size=1
00, clip_gradient=5.0, data_shape=224, gpus='0', kv_store='local', load_epo
ch=None, log_dir='/tmp/', log_file=None, lr=0.005, lr_factor=1, lr_factor_e
poch=1, model_prefix=None, network='vgg', num_classes=102, num_epochs=120,
num_examples=8189, save_model_prefix=None, train_dataset='train.rec', val_d
ataset='val.rec')
```

图 11.21　VGG-A 开始训练

```
2017-01-10 17:03:47,062 Node[0] Epoch[117] Resetting Data Iterator
2017-01-10 17:03:47,062 Node[0] Epoch[117] Time cost=175.316
2017-01-10 17:03:56,598 Node[0] Epoch[117] Validation-accuracy=0.556000
2017-01-10 17:03:56,599 Node[0] Epoch[117] Validation-top_k_accuracy_5=0.801000
2017-01-10 17:05:57,867 Node[0] Epoch[118] Batch [50]   Speed: 41.51 samples/sec     Train-accuracy=0.891600
2017-01-10 17:05:57,867 Node[0] Epoch[118] Batch [50]   Speed: 41.51 samples/sec     Train-top_k_accuracy_5=0.93900
0
2017-01-10 17:06:51,817 Node[0] Epoch[118] Resetting Data Iterator
2017-01-10 17:06:51,818 Node[0] Epoch[118] Time cost=175.219
2017-01-10 17:07:01,313 Node[0] Epoch[118] Validation-accuracy=0.567000
2017-01-10 17:07:01,313 Node[0] Epoch[118] Validation-top_k_accuracy_5=0.811000
2017-01-10 17:09:02,594 Node[0] Epoch[119] Batch [50]   Speed: 41.50 samples/sec     Train-accuracy=0.892200
2017-01-10 17:09:02,595 Node[0] Epoch[119] Batch [50]   Speed: 41.50 samples/sec     Train-top_k_accuracy_5=0.93920
0
2017-01-10 17:09:56,560 Node[0] Epoch[119] Resetting Data Iterator
2017-01-10 17:09:56,560 Node[0] Epoch[119] Time cost=175.247
2017-01-10 17:10:06,255 Node[0] Epoch[119] Validation-accuracy=0.569000
2017-01-10 17:10:06,256 Node[0] Epoch[119] Validation-top_k_accuracy_5=0.807000
```

图 11.22　VGG-A 训练结果

11.4　Torch 7 平台

11.4.1　Torch 7 平台开发环境

Torch 是一个广泛支持机器学习算法的科学计算框架，由于采用简单而快速的脚本语言和一个底层的 C/CUDA 实现，Torch 易于使用且高效，已经用在 Facebook、Google、Twitter、NYU、IDIAP、Purdue 和其他几家公司和研究实验室。

Torch 的核心是流行的神经网络和优化库，它们易于使用，同时在实现复杂的神经网络拓扑结构时具有最大的灵活性，可以建立任意的神经网络图，并在 CPUs 和 GPUs 上有效地并行化。同时，Torch 拥有一个强大的 n 维数组，很多实现索引、切片、移调的例程，还包括 LuaJIT 到 C 的接口、基于能量的模型以及线性代数、数值优化的例程。

Torch 的目标是在建立科学算法的同时，要有最大的灵活性和速度，而这一过程非常简单。Torch 拥有一个大社区驱动包的生态系统，涉及机器学习、计算机视觉、信号处理、并行



处理、图像、视频、音频和网络等，并建立在 Lua 社区基础之上。

Lua 是一个小巧的脚本语言，是巴西里约热内卢天主教大学里的一个研究小组，由 Roberto Ierusalimschy、Waldemar Celes 和 Luiz Henrique de Figueiredo 所组成并于 1993 年开发。其设计目的是为了嵌入到应用程序中，从而为应用程序提供灵活的扩展和定制功能。Lua 由标准 C 编写而成，几乎在所有操作系统和平台上都可以编译、运行。Lua 有一个同时进行的 JIT 项目，提供在特定平台上的即时编译功能。

Lua 的大部分功能来源于它的标准库，这也符合 Lua 的设计原则。因为 Lua 的主要特性就是它的可扩展性。语言中的许多特性都体现出了这点。动态类型为多态提供了支持。高阶函数和匿名函数允许实现更高层的参数化，能使函数变得更加通用。Lua 具有很多优点：

- 可扩展性，Lua 从一开始就被设计为可扩展的，既可用 Lua 代码来扩展，又可以用外部的 C 代码来扩展。
- 简易性，一个完整的 Lua 发布版本可以很轻松地存放在一张软盘中。
- 高效性，通过独立评测结果显示，Lua 是所有脚本语言中最快的语言之一。
- 可移植性，Lua 可以在任何平台上运行，不仅是 Windows 和 UNIX 平台，还包括 PlayStation、Xbox、Mac OS-9、OS X、BeOS、QUALCOMM Brew、MS-DOS、IBM 主机、RISC 操作系统、PalmOS、ARM 处理器、Rabbit 处理器、类 UNIX、类 Windows 系统。

同样，Torch 7 既可以安装在 Windows 系统上，也可以安装在 Linux 操作系统上，安装过程相对简单，便于操作。本章的实验是在基于 Ubuntu 14.04 版本的操作系统下进行的。另外，由于 Face book 开源了他们基于 Torch 的深度学习库包，这个版本包括 GPU 优化的大卷积模块，以及稀疏网络，这些通常被用在自然语言处理中的应用中。使用者可以在 Torch 平台上更自由地实现对已有模块逻辑复杂的调用，更加简单地实现自己的算法，不用浪费精力在计算优化上面。

11.4.2 二值神经网络

二值神经网络(Binary Neural Network，BNN)起源于 1943 年 Warren McCulloch 和 Walter Pitts 提出的人工神经元模型，真正兴起是在 2015 年底，由 Yoshua Bengio 领导的蒙特利尔大学的研究组在 arxiv.org 上发表的文章——*Binarized Neural Networks with Weights and Activations Constrained to* +1 *or* −1，如图 11.23 所示。二值神经网络是神经网络“小型化”探索中重要的一个方向。针对传统的神经网络模型，网络的系数以及网络的中间结果可作为二值化部分。通过把浮点单精度的系数变成正 1 或负 1，系数的二值化能达成存储大小变为原来的 1/32，也就是 3%。另外，由于大部分计算都在−1 和 1 之间进行，可以将浮点计算替换成整数位计算。在支持 64 位运算的 CPU 和 GPU 上，这意味着 64 倍的理论加速比。

二值神经网络的主要思想是通过二值化权值参数 $\boldsymbol{W}$ 和隐藏层激活值 $\boldsymbol{a}$，来减少存储内

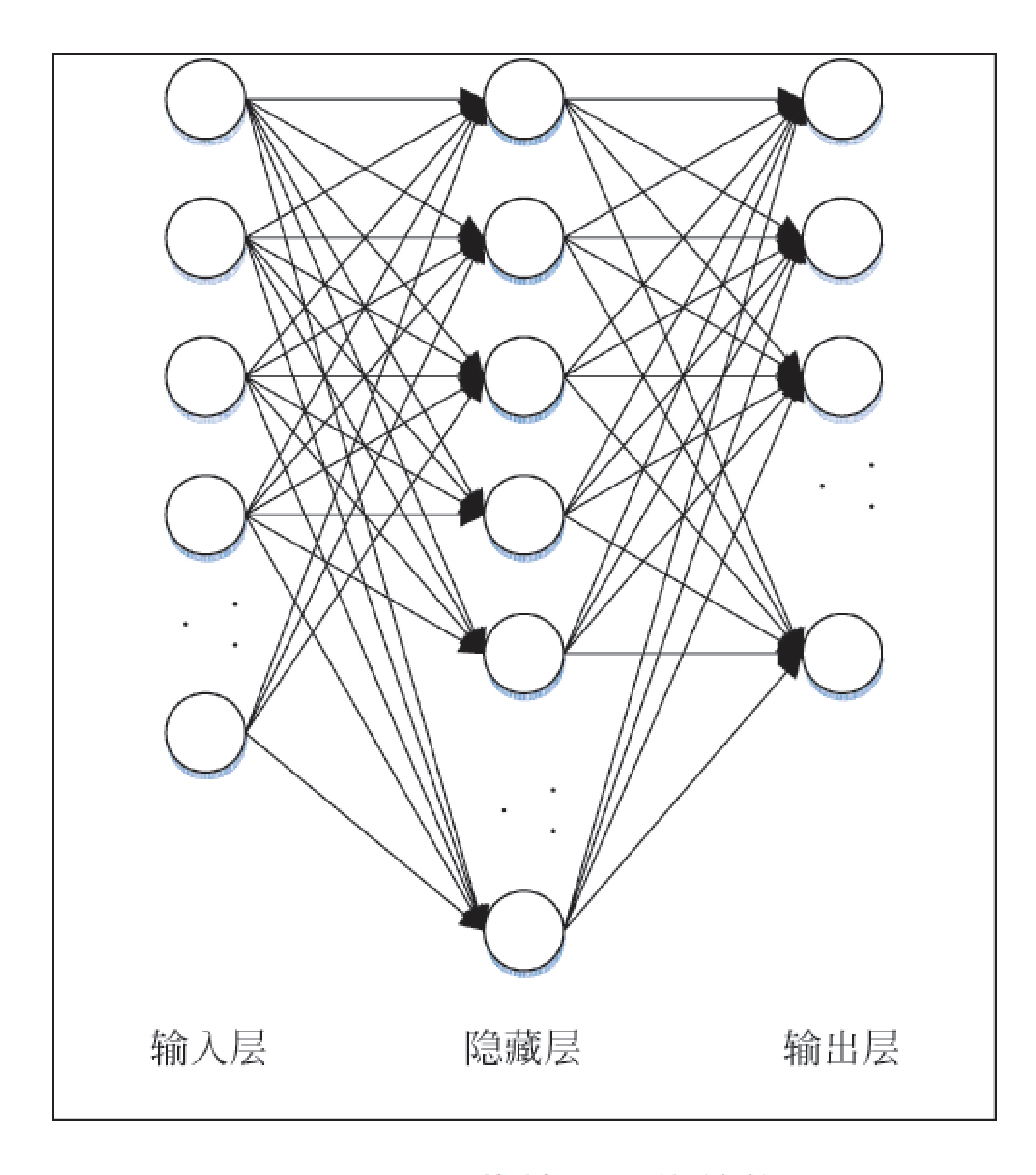


图 11.23　二值神经网络结构图

存的占用。同时利用位移操作来代替网络中的乘法运算，大大减少了运算时间。二值网络训练时的权值参数 $\boldsymbol{W}$，必须包含实数型的参数，然后将实数型权值参数二值化，得到二值型权值参数，即

$$\boldsymbol{W}_k^b = \mathrm{Binarize}(\boldsymbol{W}_k) \tag{11.9}$$

然后利用二值化后的参数计算得到实数型的中间向量，该向量再通过 Batch Normalization 操作，得到实数型的隐藏层激活向量。如果不是输出层，就将该向量二值化。

通常，采取的二值化方法为：正数置为 1，负数置为 −1，如图 11.24 所示。

$$x^b = \mathrm{sign}(x) = \begin{cases} +1 & x \geqslant 0 \\ -1 & \text{其他} \end{cases} \tag{11.10}$$

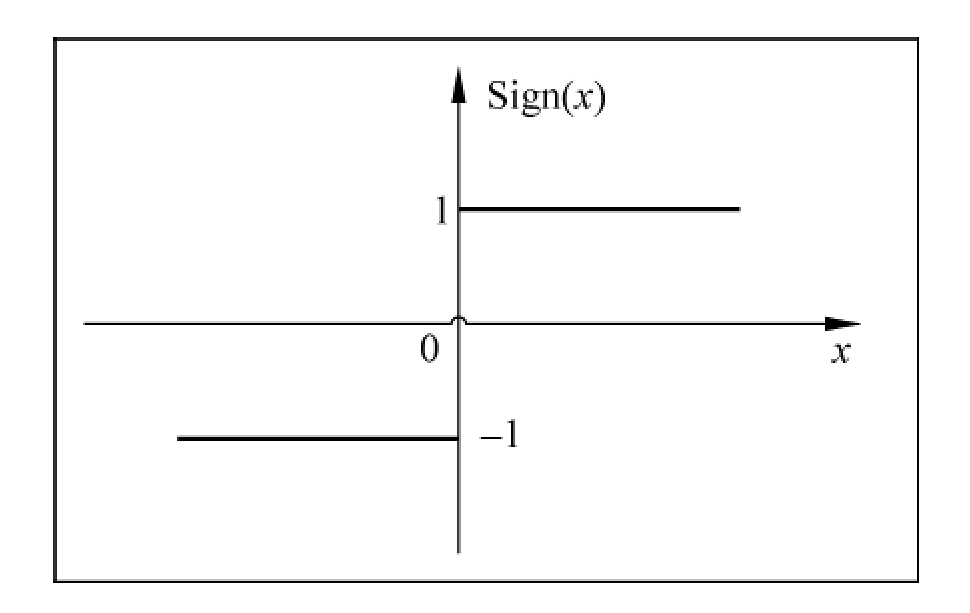


图 11.24　sign(x)函数

但是，考虑到符号函数不好进行梯度传播的反向传播，因为如果直接对决定式的二值化函数求导，求导后的值均为零。所以采取一种妥协方法，将 sign(x)进行宽松，实现函数的可导。如图 11.25 所示，改进后的函数为：

$$\mathrm{Htanh}(x)=\mathrm{Clip}(x,-1,1)=\max(-1,\min(1,x)) \tag{11.11}$$

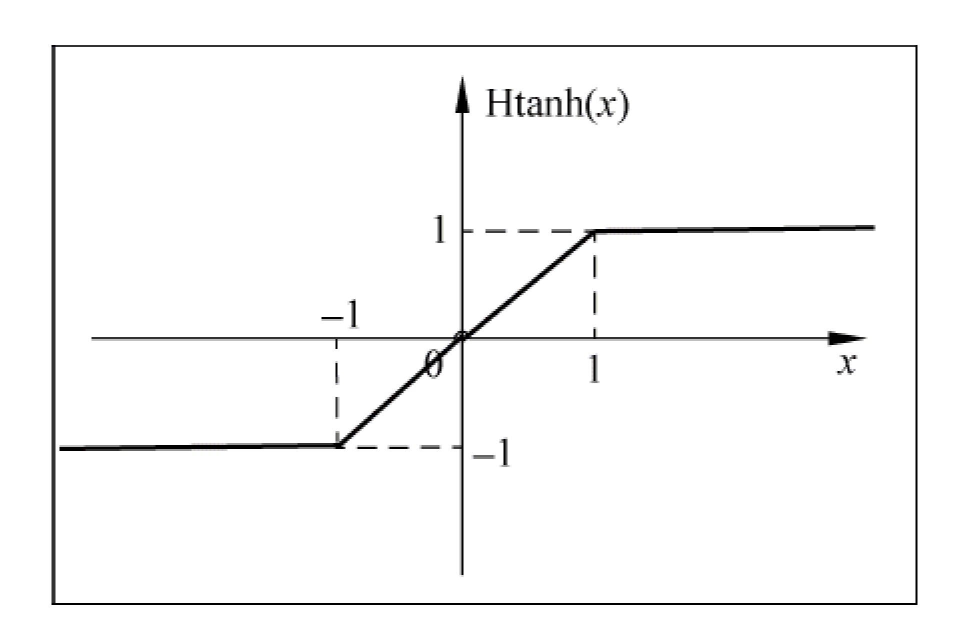


图 11.25　Hard tanh(x)函数

在求梯度时，根据链式法则，在求解第 k 层和第 $k+1$ 层的权值参数的梯度之前，必须先求解第 $k+1$ 层的误差值。即

$$W_{ij}^{(l)}=W_{ij}^{(l)}-\alpha\cdot\frac{\partial J(\boldsymbol{W},\boldsymbol{b})}{\partial W_{ij}^{(l)}} \tag{11.12}$$

其中：

$$\frac{\partial J(\boldsymbol{W},\boldsymbol{b})}{\partial W_{ij}^{(l)}}=\frac{1}{m}\sum_{i=1}^{m}\frac{\partial}{\partial W_{ij}^{(l)}}J(\boldsymbol{W},\boldsymbol{b};\boldsymbol{x}^{(i)},y^{(i)})+\lambda\cdot W_{ij}^{(l)} \tag{11.13}$$

$$\frac{\partial J(\boldsymbol{W},\boldsymbol{b};\boldsymbol{x},y)}{\partial W_{ij}^{(l)}}=a_j^{(l)}\cdot\delta_i^{(l+1)} \tag{11.14}$$

在求梯度时，需注意由于二值网络中，除了输出层，其他隐藏层都需经过二值化，所以在求 Batch Normalization 的参数时，必须先求二值操作层。二值网络在对权值求梯度时，是对二值化后的权值求梯度，而不是对二值化前的实数型权值求梯度，因为二值化前的权值没有真正地参与网络的前向传播过程。另外，在权值更新过程中，与求权值梯度不同，不再是对二值化后的结果更新，而是利用实数型的权值进行更新。最后，是关于乘法的优化，对 Batch Normalization 的优化主要通过 AP2(x)和简单的位移操作，AP2(x)的作用是求与 x 最接近的 2 的幂次方。

11.4.3　二值神经网络应用于图像分类

在本章的实验中，主要将二值神经网络用于 MNIST 数据集和 SVHN 数据集的分类。

MNIST 是一个手写数据的数据库，它有 50 000 个训练数据集和 10 000 个测试数据集。MNIST 的图像块大小是 28×28，要实现的是 10 分类问题。

实验中的模型部分为 4 层全连接的网络，如图 11.26 所示。各层的节点数分别为 784、

2048、2048、2048、10。最后一层采用 hinge loss 作为损失函数，hinge loss 常用于"maximum-margin"的算法。实验共训练了 100 代，BatchSize 设为 100。实验过程对应的二值神经网络具体模型如下：

实验过程中，MNIST 训练数据集的最终准确率为 99.95%，测试数据集的准确率为 98.43%。图 11.27 为 MNIST 训练数据的混淆矩阵，图 11.28 为 MNIST 测试数据的混淆矩阵，图 11.29 为 MNIST 在二值神经网络训练和测试过程中的错误率折线图。

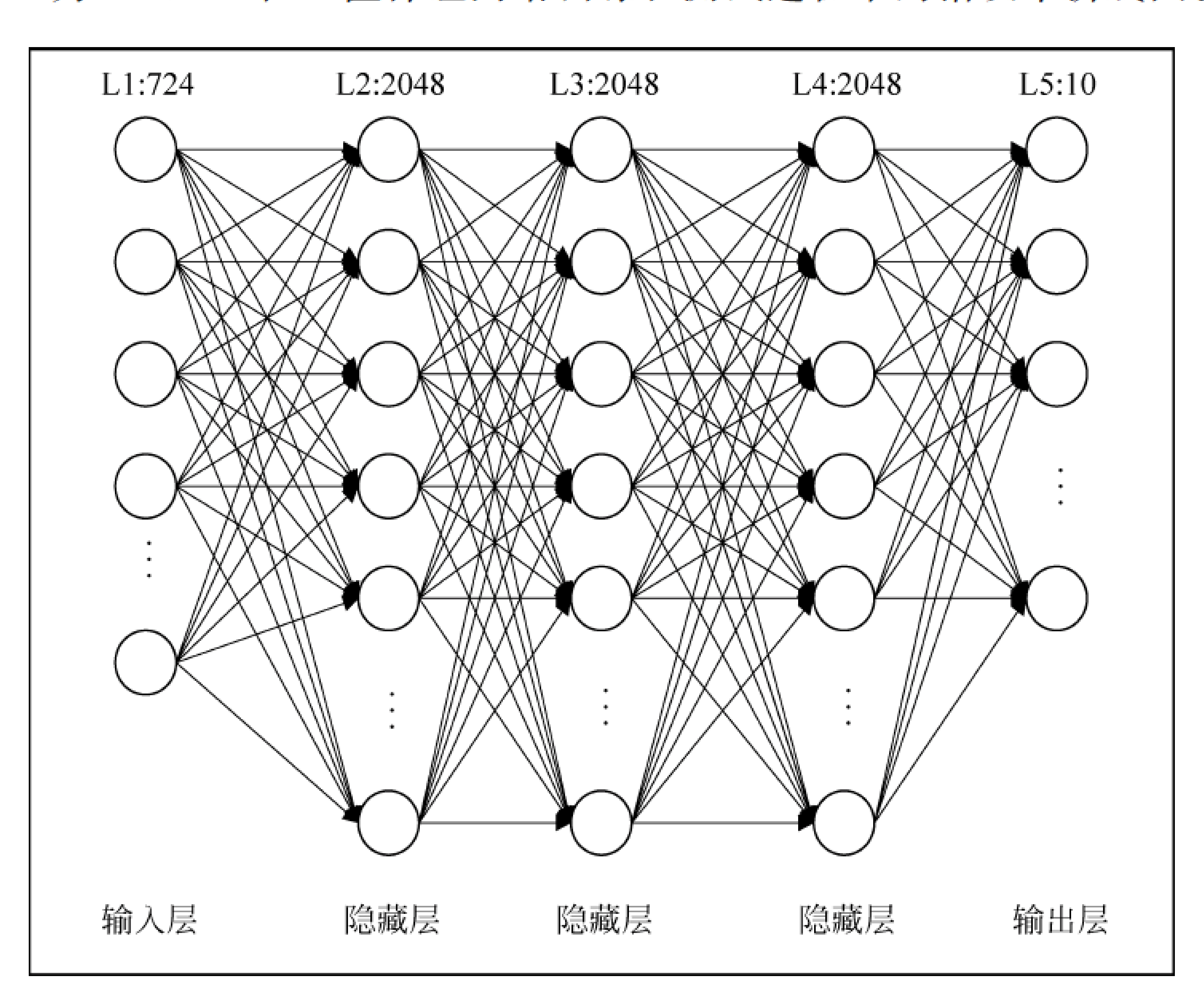
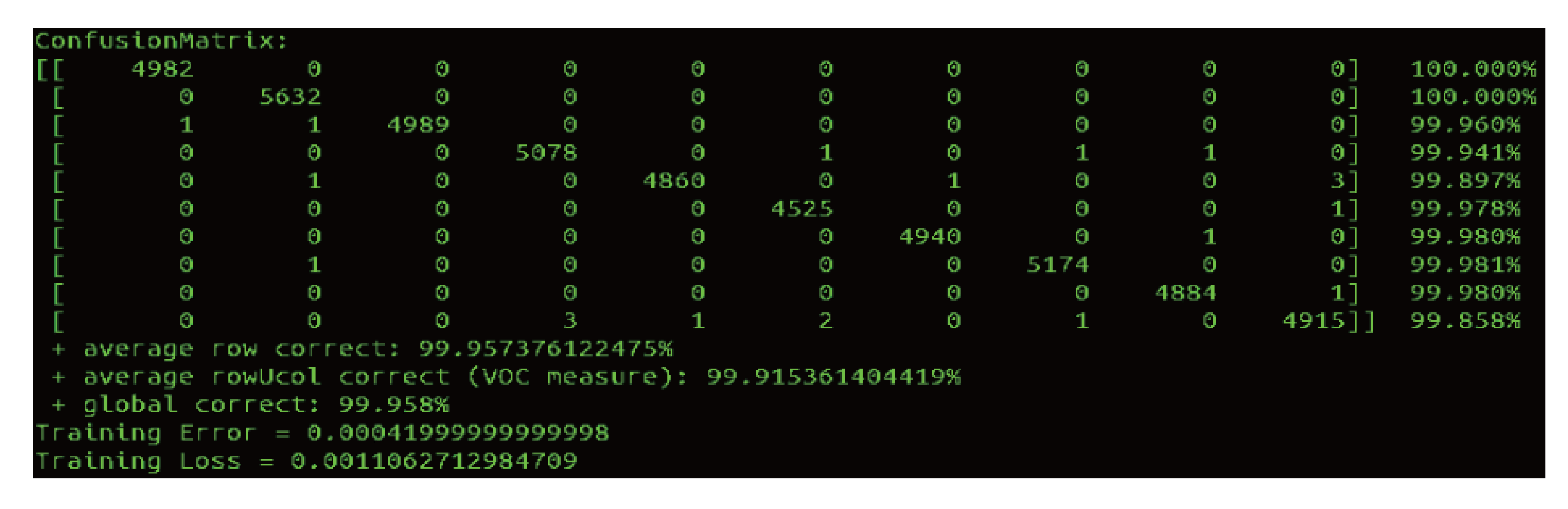


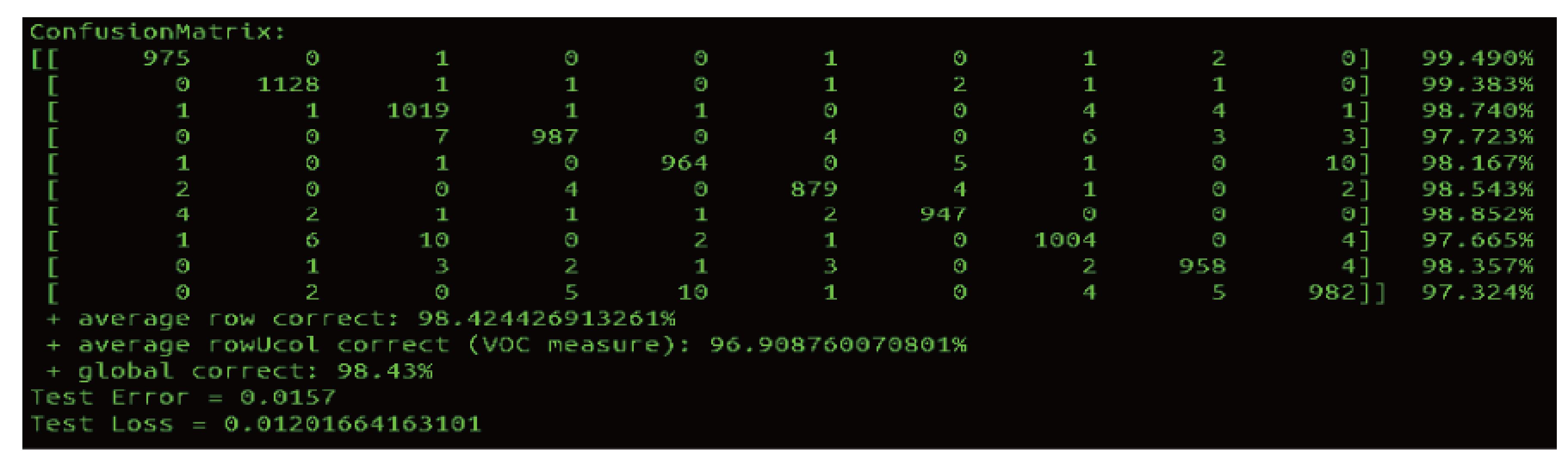
图 11.26　MNIST 对应的二值神经网络模型



```
ConfusionMatrix:
[[   4982      0      0      0      0      0      0      0      0      0]   100.000%
 [      0   5632      0      0      0      0      0      0      0      0]   100.000%
 [      1      1   4989      0      0      0      0      0      0      0]   99.960%
 [      0      0      0   5078      0      1      0      1      1      0]   99.941%
 [      0      1      0      0   4860      0      1      0      0      3]   99.897%
 [      0      0      0      0      0   4525      0      0      0      1]   99.978%
 [      0      0      0      0      0      0   4940      0      1      0]   99.980%
 [      0      1      0      0      0      0      0   5174      0      0]   99.981%
 [      0      0      0      0      0      0      0      0   4884      1]   99.980%
 [      0      0      0      3      1      2      0      1      0   4915]]  99.858%
 + average row correct: 99.957376122475%
 + average rowUcol correct (VOC measure): 99.915361404419%
 + global correct: 99.958%
Training Error = 0.00041999999999998
Training Loss = 0.0011062712984709
```

图 11.27　MNIST 训练数据的混淆矩阵

SVHN 是一个街道门牌号数字识别数据集，是一个 10 分类问题，主要用于机器学习以及目标识别算法中。SVHN 与 MNIST 的不同在于：MNIST 是单通道的灰度图像，而 SVHN 是 3 通道的自然图像，且 SVHN 的图像大小为 32×32。在本章的实验中，训练集大小为 73257，测试集大小为 26032。实验过程对应的二值神经网络具体模型如图 11.30 所示。



```
ConfusionMatrix:
[[     975       0       1       0       0       1       0       1       2       0]   99.490%
 [       0    1128       1       1       0       1       2       1       1       0]   99.383%
 [       1       1    1019       1       1       0       0       4       4       1]   98.740%
 [       0       0       7     987       0       4       0       6       3       3]   97.723%
 [       1       0       1       0     964       0       5       1       0      10]   98.167%
 [       2       0       0       4       0     879       4       1       0       2]   98.543%
 [       4       2       1       1       1       2     947       0       0       0]   98.852%
 [       1       6      10       0       2       1       0    1004       0       4]   97.665%
 [       0       1       3       2       1       3       0       2     958       4]   98.357%
 [       0       2       0       5      10       1       0       4       5     982]]  97.324%
 + average row correct: 98.424426913261%
 + average rowUcol correct (VOC measure): 96.908760070801%
 + global correct: 98.43%
Test Error = 0.0157
Test Loss = 0.01201664163101
```

图 11.28　MNIST 测试数据的混淆矩阵

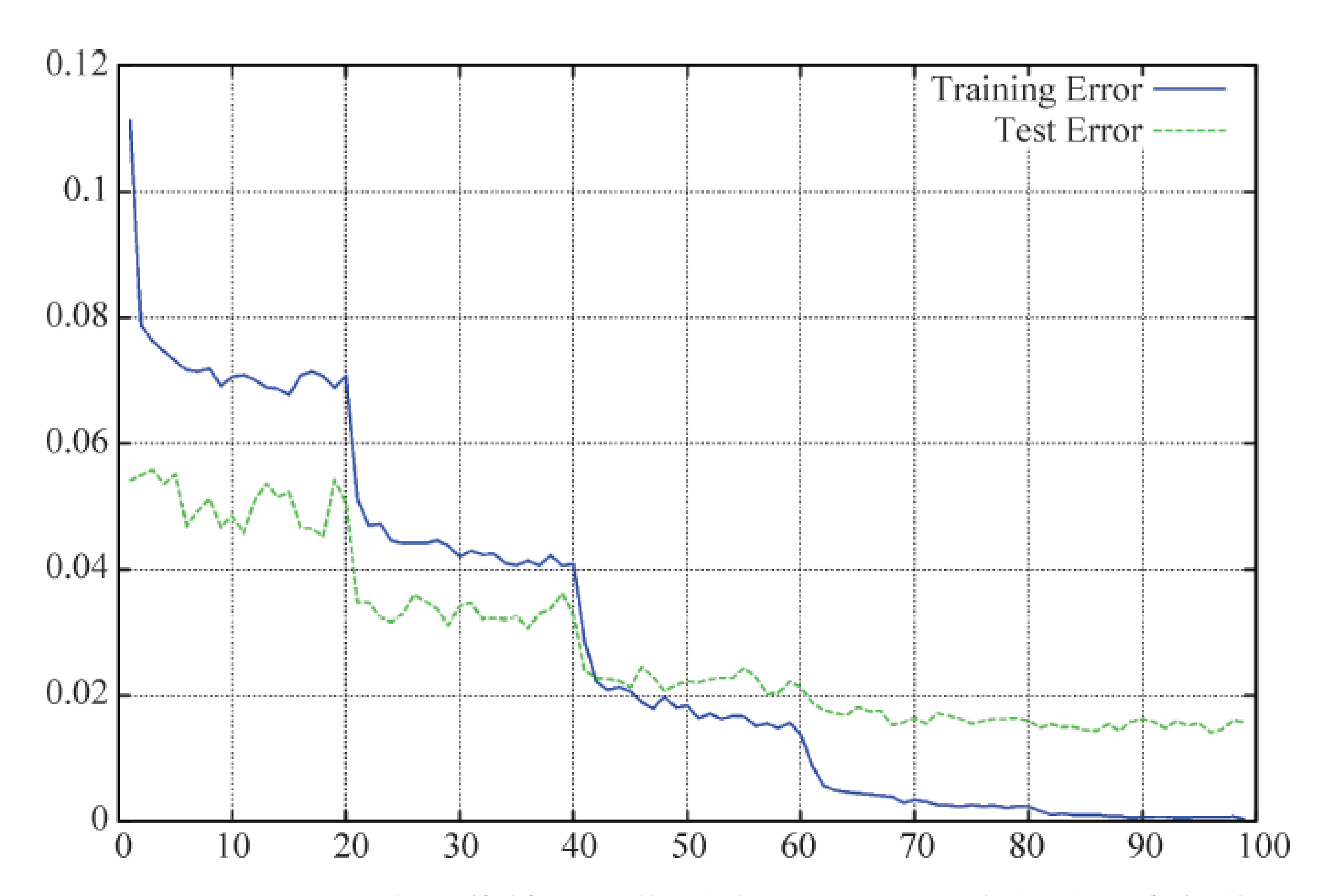


图 11.29　MNIST 在二值神经网络训练和测试过程中的错误率折线图

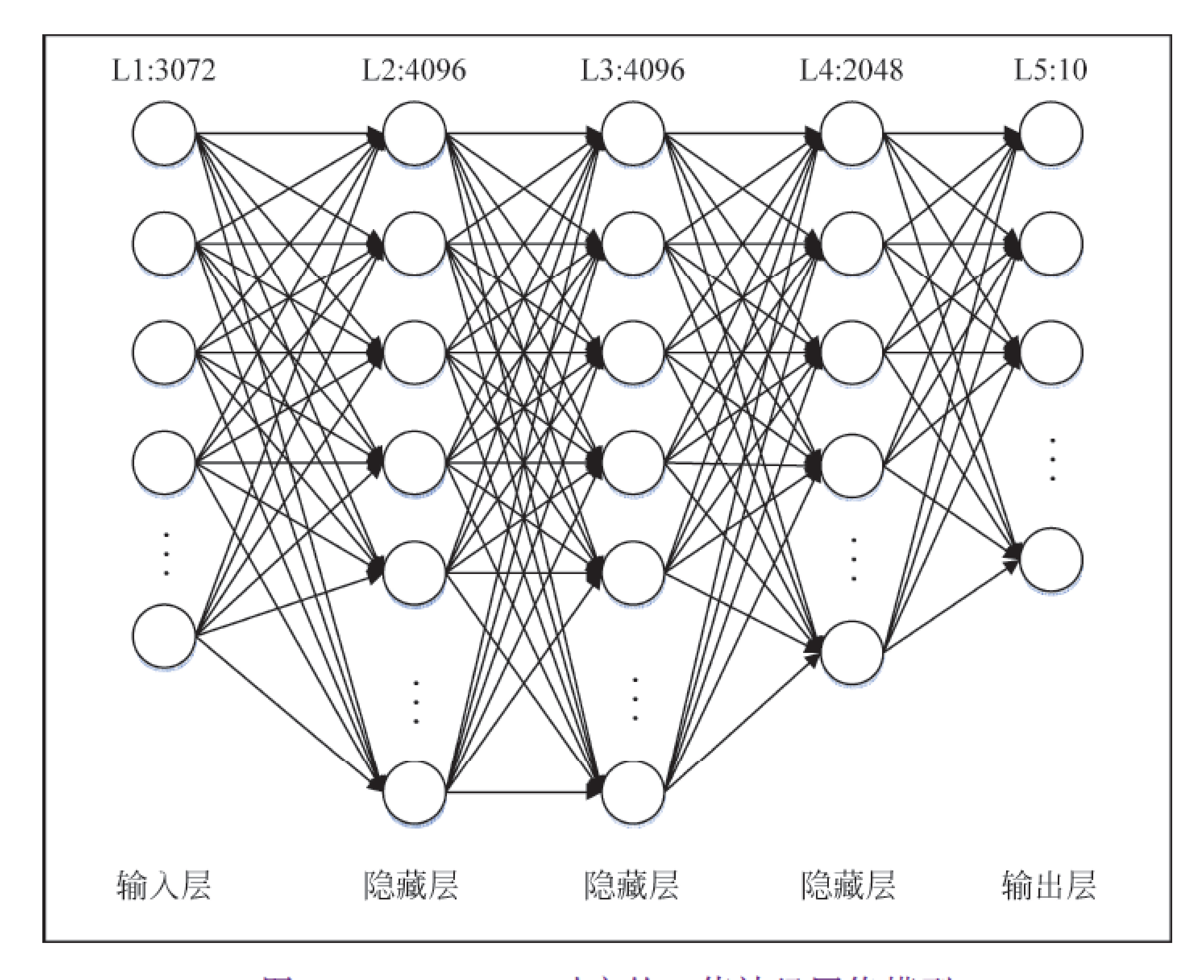


图 11.30　SVHN 对应的二值神经网络模型

实验统计，SVHN 训练数据集的最终准确率为 98.38%，测试数据集的准确率为 84.17%。图 11.31 为 SVHN 训练数据的混淆矩阵，图 11.32 为 SVHN 测试数据的混淆矩阵，图 11.33 为 SVHN 在二值神经网络训练和测试过程中的错误率折线图。

```
ConfusionMatrix:
[[  13745     20     18     26      6      9     11     11      7      8]  99.163%
 [     30  10470     15     12     12      6     19      8     11      2]  98.914%
 [     24     26   8345     10     34      8     14     13     16      7]  98.211%
 [     25     11     13   7373      2      7      8      3     10      6]  98.860%
 [     24     11     29      6   6752     27      4      9      9     11]  98.111%
 [     19     14     11     11     12   5608     11     18      9     14]  97.922%
 [     26     14     13     11      7      6   5498      8      4      8]  98.266%
 [     23     15     17     15     12     19     10   4904     12     18]  97.205%
 [     23     22     23     10     21      5     12     15   4511     17]  96.823%
 [     23     12      8      5      5      6      6      9      6   4868]]  98.383%
 + average row correct: 98.185923099518%
 + average rowUcol correct (VOC measure): 96.590624451637%
 + global correct: 98.385137256508%
Training Error = 0.016148627434921
Training Loss = 0.022854670896648
```

图 11.31　SVHN 训练数据的混淆矩阵

```
ConfusionMatrix:
[[   4707     74     64     41     34     25     59     41     39     15]  92.312%
 [     98   3623    101     32     65     25     69     48     81      7]  87.322%
 [    149     69   2185     19    135     14     15     47    238     11]  75.815%
 [    109     53     32   2163     37     21     10     30     62      6]  85.731%
 [     53     33    104     21   1990     44     14     67     50      8]  83.473%
 [     63     21     27     41     87   1574     10     90     35     29]  79.616%
 [    136     75     31      4     15     16   1693     14     33      2]  83.853%
 [     51     17     30     22     52     55     11   1323     86     13]  79.699%
 [     41     50     26     13     46     17     13     29   1337     23]  83.824%
 [     57     41     50     12     35     60     24     50     98   1317]]  75.516%
 + average row correct: 82.716256976128%
 + average rowUcol correct (VOC measure): 71.047068238258%
 + global correct: 84.173325138291%
Test Error = 0.15826674861709
Test Loss = 0.10921624560939
```

图 11.32　SVHN 测试数据的混淆矩阵

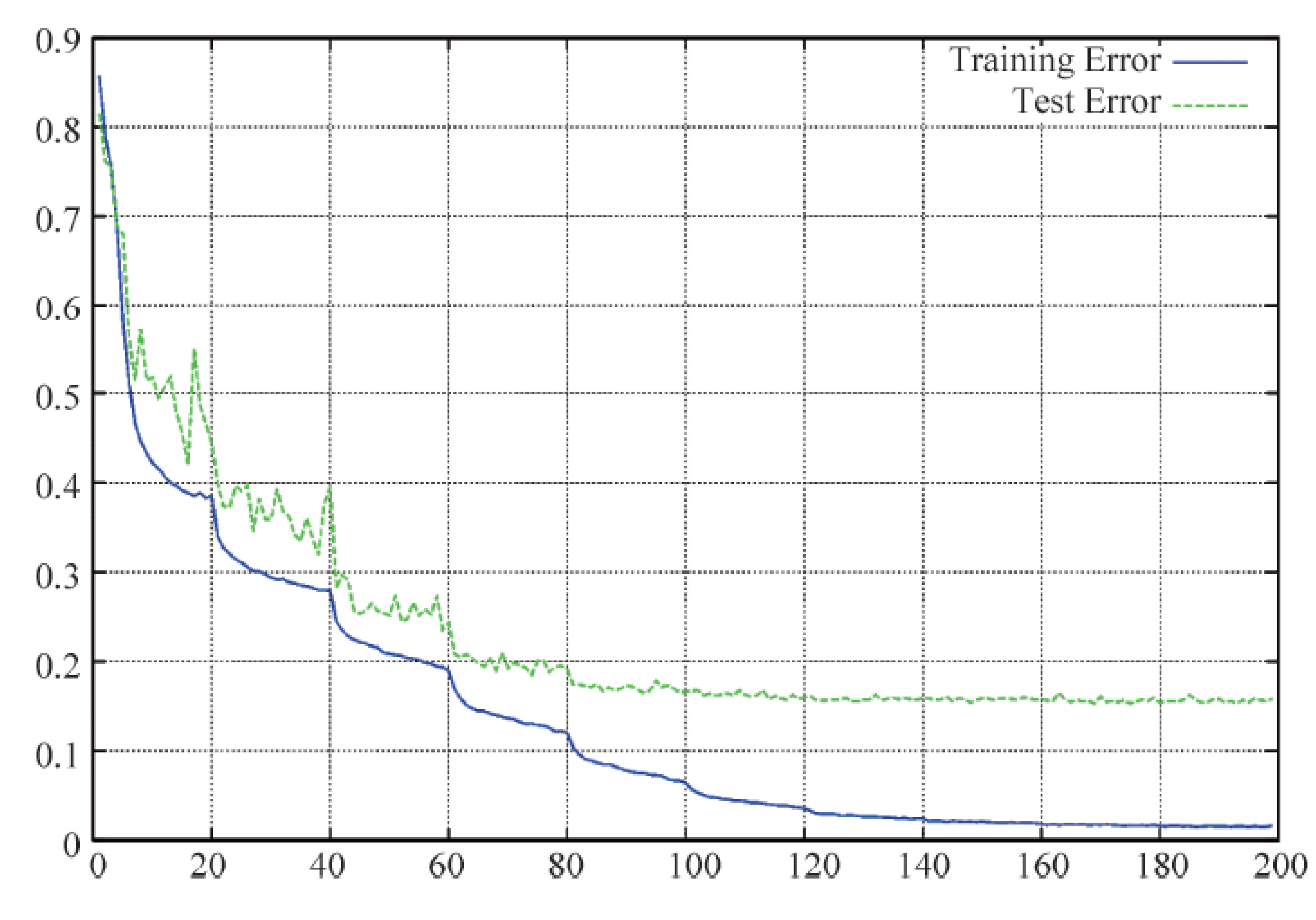
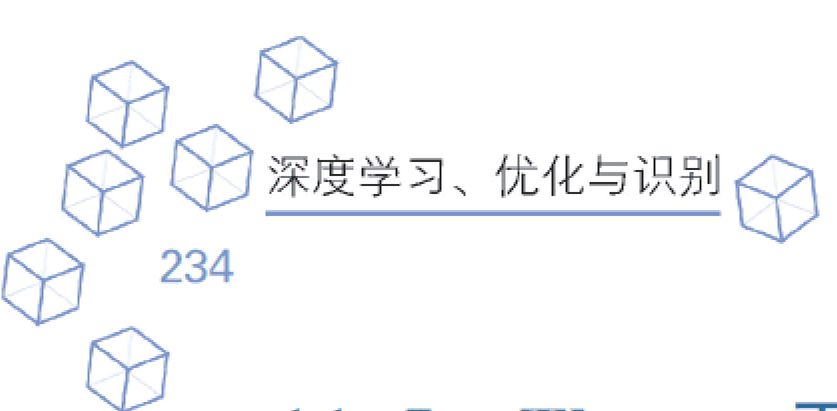


图 11.33　SVHN 在二值神经网络训练和测试过程中的错误率折线图



11.5 Theano 平台

11.5.1 Theano 平台开发环境

Theano 由 LISA 实验室在蒙特利尔大学开发，是深度学习较早的库之一，由深度学习三大先驱（Hinton、LeCun、Bengio）中的 Bengio 构建。用来支持机器学习算法的快速发展，专门为深度学习中大型神经网络的计算而设计，可以进行多种深度学习模型的训练和架构。Theano 是 BSD 许可证发布的开源软件，包含许多自定义 C 和 CUDA 代码生成器，可以适应多种格式和大小的输入。它使用 Python 编写，C 编译执行，CUDA 并行加速，可以安装在 Linux 或 Windows 系统上，需要 Python 环境支持。它支持 GPU 加速，支持 cuDNN 深度学习库等。Theano 不是正常意义上的编程语言，而是使用 Python 编写程序生成 Theano 的表达式。Theano 实际是一个 Python 库，允许定义、优化和计算数学表达式，特别是提高多维数组的运算性能。

另外，对于大数据问题，使用 Theano 相对于 C 语言具有速度上的优势，它也可以利用 GPU，在性能上大大超越运用 CPU 的 C 语言。Theano 将计算机代数系统（CAS）和一种优化编译器结合。它还可以为许多数学运算生成自定义的 C 代码。当复杂的数学表达式重复进行计算并且评价速度很关键时，CAS 与优化编译这个组合尤为重要。在许多不同表达每个都被评价一次的情况下，Theano 可以减少汇编和分析的开销，但仍然提供符号功能，如自动分化。Theano 是一个数学表达式的编译器，其编译器为符号表达式提供复杂程度不同的优化。这些优化包括但不是限于：使用 GPU 计算；常量叠算；支持张量和稀疏表达；支持线性代数计算；惰性求值；并行计算；符号微分；支持大部分 Numpy，基本的 Scipy 函数；支持图变换，包括：微分/高阶微分，'R'和'L'微分算子，速度/记忆优化，数值稳定性优化；支持 CUDA 后端；可以循环工作。Theano 可以使用 g++或 nvcc 编译表达图为 CPU 或 GPU 的指令，比单一 Python 运行得更快；可以自动建立符号图用来计算梯度；可以识别一些不稳定的数值表达并用更稳定的数值运算来计算。Theano 更关注张量表达，编译更加机械化。Theano 内部构造为一个图结构，包含变量节点、操作节点、应用节点。其中应用节点代表对一些变量的操作。因为它是一个代数符号系统，所以数学表达式中的符号就是 Theano 的变量，数学变量用运算符等操作符连接起来，就形成 Theano 的一个图。如果需要构建神经网络，本质就是建立一张大图。拥有图结构，使得微分计算非常简单，每张图的结构都清晰地展示了从输入到输出的运算过程，从而使得提高计算方式成为可能。Theano 中的优化一般都是将现有的图或子图用一些可以达到相同的计算结果，却更加稳定高效地图来代替。Theano 构建神经网络十分方便，提供几个基础的神经网络模型代码，并且含有详细的注释。使用 Theano 搭建神经网络是因为它可以自动计算梯度，并且只需要定义函数和计算梯度两个过程。其构建的深度学习代码一般包含四个部分：数据、模型、预训练和训练、测试。另外有许多基于 Theano 的深度学习库，如 lasagna、keras、nolearn，使用这些

库，可以通过简单的堆叠构建一个完整的神经网络，为我们的学习提供很大的便利。

11.5.2　递归神经网络

神经网络是模拟大脑皮层的运作机理，建立神经元之间的联系，用来获取学习、理解信息。传统的神经网络是输入层到隐藏层，隐藏层到输出层之间的全连接，而实际上我们的大脑的连接要更加复杂，不只层与层之间，每层内部也应该有连接，当理解一篇文章，或者识别一个乐谱时，我们不只需要当前知识，还要参考之前积累的信息，针对这些问题，RNN 网络应运而生(见图 11.34)，其特点就是隐藏层内部各个神经元之间有着循环的传递回路。

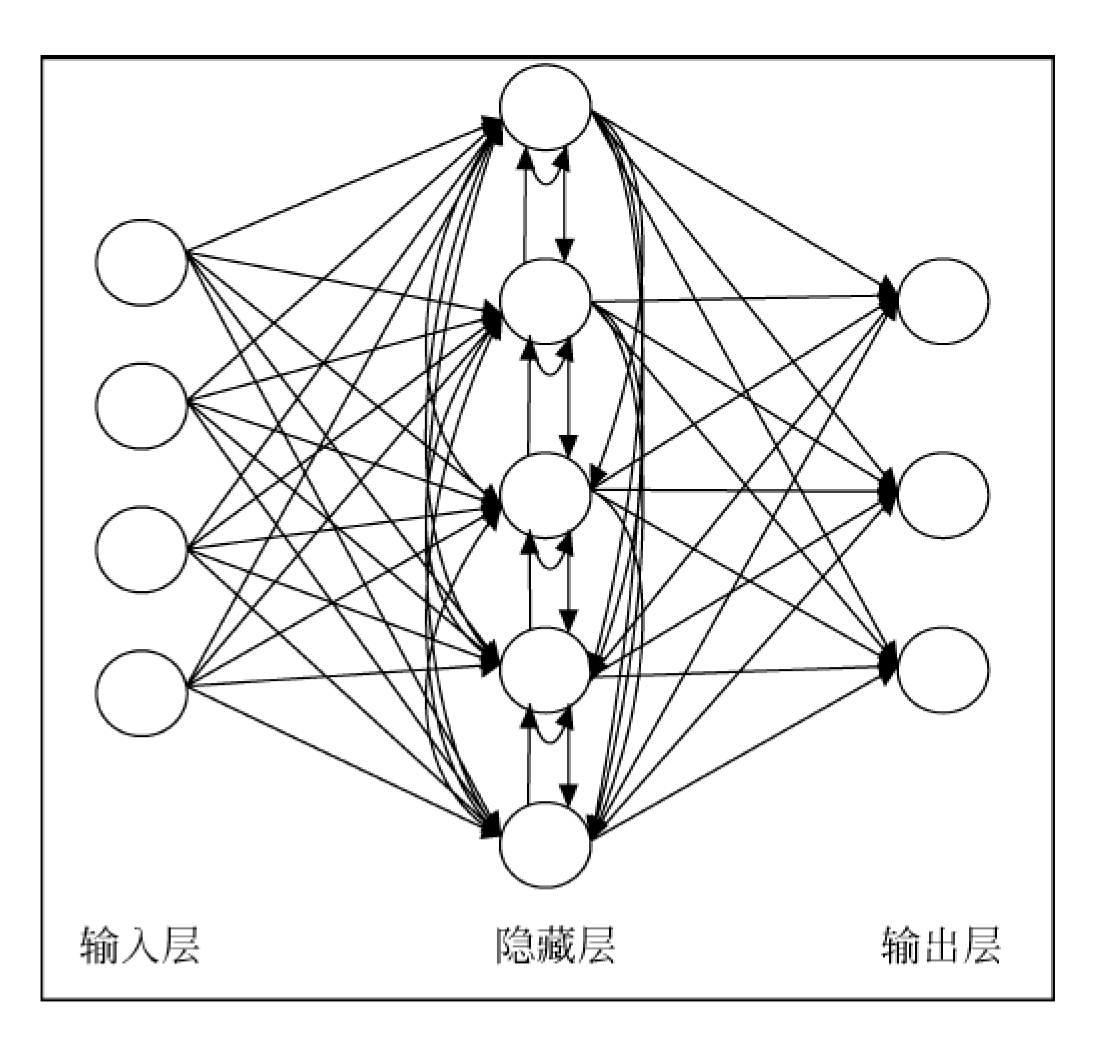


图 11.34　递归神经网络(RNN)结构图

RNN 具备了对时间的建模功能，可以存储上一时刻的信息，用来推断下一时刻的分类结果，即神经网络的输出是当前时间输入(输入层的输出)与上一时刻输出(隐藏层的输出)共同作用的结果，隐层之间存在闭环回路，使得从任意时刻出发，可以接收上一时刻的信息，故将其称为递归神经网络。RNN 具有记忆能力，与时间有关，它与普通神经网络不同的是，一个隐层可视为多个隐层，且隐层节点之间互有连接，因此可以将 RNN 按照时间序列展开，如图 11.35 所示。从图 11.35 可以看出，RNN 隐层可以看作将几个普通神经网络拼接起来。

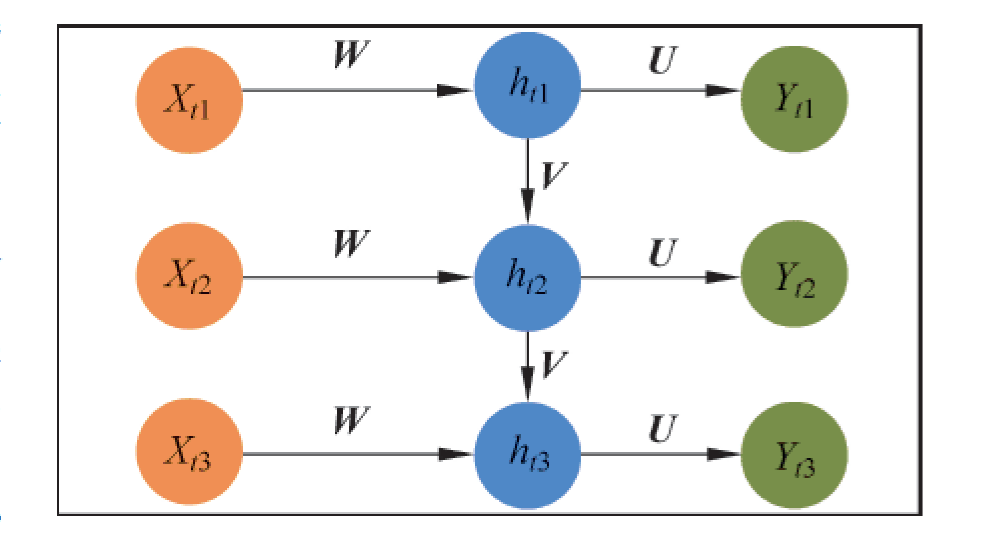


图 11.35　RNN 隐藏层按时间展开图

RNN 的输入一般为一个时间序列，在该图中即 $X=[X_{t1}, X_{t2}, X_{t3}]$；若输入 X 为一句话，则 X_{tn} 代表一个单词，编程的时候，必须将单词转换为计算机可以识别的形式，所以要将一个单词表示为一个向量，即词向量，可以用一

个单位向量来表示。h_{tm}代表隐藏层的输出，即网络所记忆的内容，一般是输入与权重经过非线性激活函数得到的值，在 RNN 中，输入要包含输入与输入层和隐藏层之间的权重 W 以及上一时刻隐藏层输出和隐藏层节点的权重 V 两部分。Y_{tm}代表网络输出，它仅与当前时刻隐藏层的输出有关，一般可经过 softmax 分类器得到。

RNN 与普通神经网络另一点不同是权值共享。每输入一个序列，从该序列输入，到隐藏层，再到输出，每一层拥有同样的参数，这使得学习的关联性更强，学习性能更好，并且降低了网络需要学习的参数。RNN 的训练过程是：首先将一个序列按时间分为许多子序列，每次送入网络一个子序列，将当前时刻子序列与隐藏层经过计算得到当前时刻的隐藏层的输出值，再与上一时刻隐藏层的输出值联合，共同送入输出层，得到输出结果。接下来再对下一时刻的子序列做相同的运算。可以将 RNN 类比为一个权重都一样的前馈神经网络，其训练过程体现了其记忆功能，这使其在文本、语音等的处理方面具有很大的优势。

RNN 特有的循环特征，使得其可以寻找历史和未来之间的关系，可以将它看作推断机器。RNN 在文本生成、语言模型、机器翻译等方面有很突出的贡献。序列数据预测是机器学习和人工智能的重要问题之一，语言模型的目的就是根据给定文章中的文字数据预测下一个文字，因此生成语言模型就是要考虑预测序列数据。适用于语音识别和机器翻译的语言模型需要大量的数据，已有的许多模型都过于复杂且仅适用于少量的数据。例如 Bengio 提出的运用前馈神经网络统计语言模型，只能用固定长度的文本，通常预测下一个单词需要之前 5～10 个单词。而对于 RNN 的循环结构，不受文本长度的限制，信息在网络内部循环，实现一定时间内的记忆，这对于现实生活中长短不一的语句的预测更加准确。

RNN 做图像描述生成就是生成一幅图片和图片区域对应的文字描述，模型的目的是生成语言和可视化数据之间的关系，使用的数据库包含许多图片，且每幅图片有其对应的语言描述，模型训练时输入图片和语言描述，在测试阶段就可以利用训练好的模型使输入的图片输出其对应的文本描述。训练过程中将语句的单词转为向量，作为序列输入 RNN 中，RNN 再根据这个词和上一个词来预测下一个词，最终预测出图片对应的语句。

RNN 做音乐生成最直接的方式是将网络作为单步预测器，网络利用 t 时刻的信息作为输入来预测 $t+1$ 时刻的信息，整个学习结束后，网络就根据初始输入和由隐层生成的子序列的输入，生成新的信息。对于前馈神经网络，不具备存储过去信息的能力，便不能模拟一首音乐的旋律。做音乐生成，需要将音乐数据转为序列数据，根据读入的音符，学习出一段旋律。

RNN 属于深度神经网络，相对于普通神经网络，其深度不只是对于隐藏层的层数，还指隐藏层展开的时间步数的深度，RNN 利用权值共享，当前步的输出是之前很多步共同作用的结果，所以在网络训练的过程中，很容易出现梯度消失，尤其是对于长时间的记忆，RNN 就不能很好地运作。针对 RNN 的不足，发展了一系列改进的 RNN，其中典型的有双向循环网络(Bidirectional RNNs，BRNN)、长短时记忆网络(Long Short Term Memory，LSTM)等。

Bidirectional RNNs(见图 11.36)在 1997 年由 Schuster 和 Paliwal 发明。它不只可以

访问上一时刻的信息，还可以加入未来的信息，当我们要预测一个语句中中间部分的词语时，需要考虑其前后的因素，因此需要双向循环网络。其每一个展开的序列都有前后两个循环回路。网络的输出由两个回路的隐层输出共同决定。

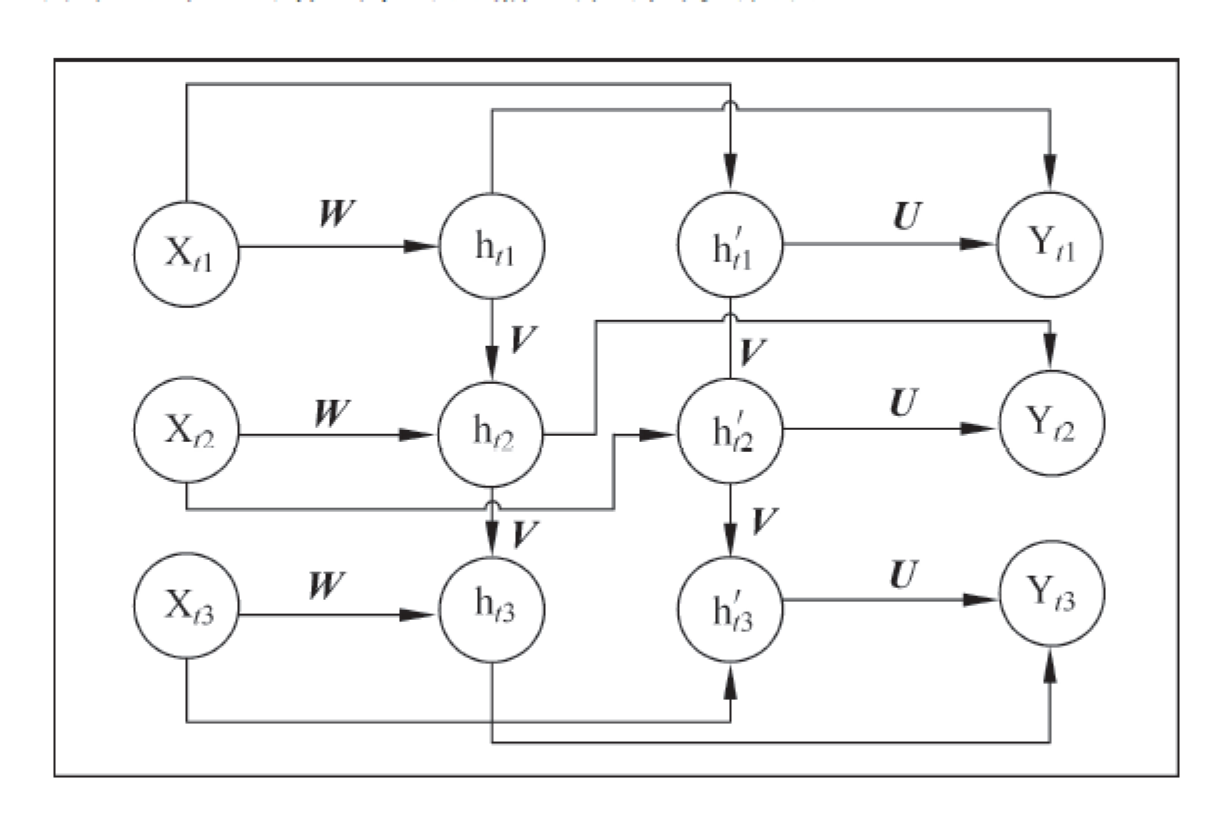


图 11.36　BRNN 隐藏层按时间展开图

LSTM 在 1997 年由 Sepp Hochreiter 和 Jurgen Schmidhuber 提出，其结构如图 11.37 所示，是针对 RNN 不能进行长时间的记忆而改进的网络。其主要思想是通过门将短时记忆的信息存储起来。前面讲过 RNN 按照时间展开可以看作几个普通的单隐层神经网络，LSTM 就是将隐层变为四层。当写入门的权重为 1 时，允许循环读取外部网络的内容，当保持门的权重为 1 时，会将内容记忆在循环中，当保持门权重为 0 时，记忆将会被清除，当读取门的权重为 1 时，允许外部网络从循环中读取内容。这样当网络需要某一记忆值时，就将该值输出，若不需要该值就在循环体内部循环，直至消除该记忆。这样从该值读取到写入，通过所有的单元后权重约为 1，产生的反向传播误差可忽略，所以有效地避免了梯度弥散现象。

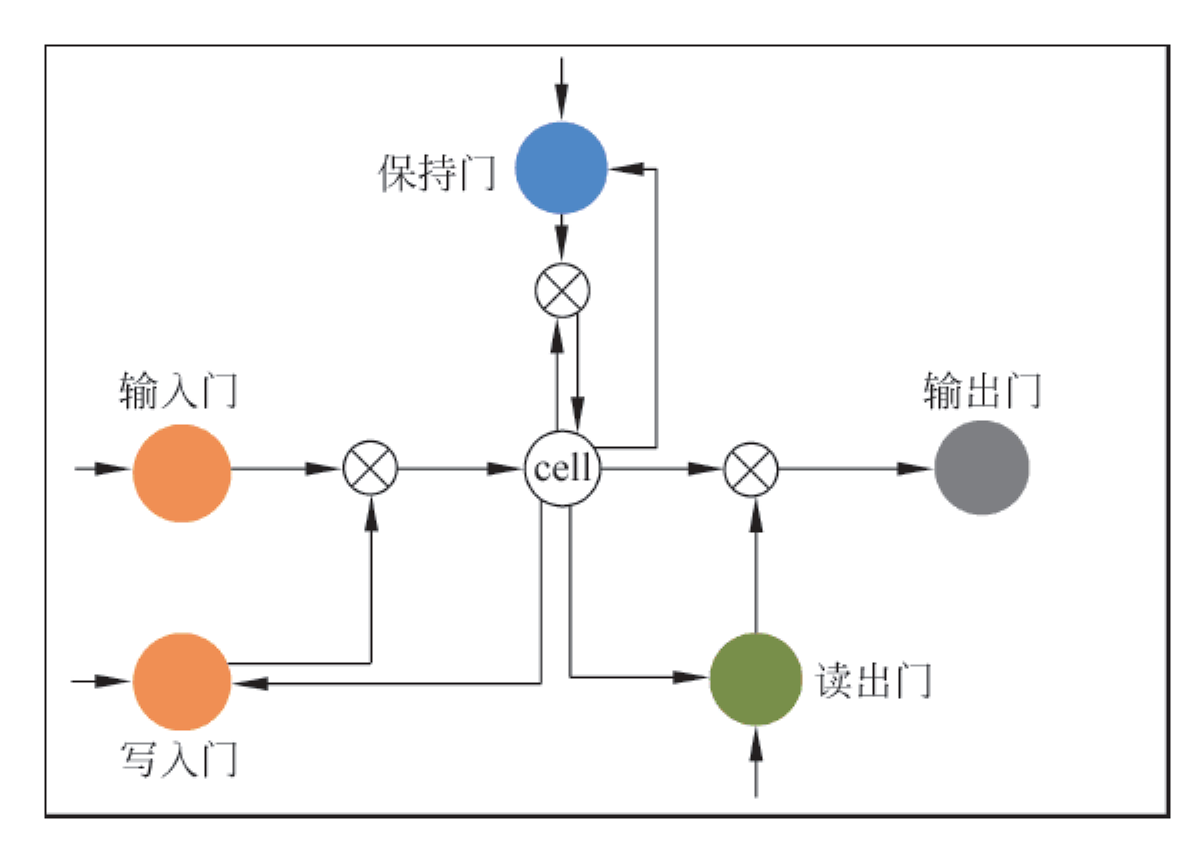


图 11.37　LSTM 网络结构图

11.5.3 LSTM 应用于情感分类任务

1. 数据集介绍

IMDB 数据集是目前互联网最权威的电影数据集。数据集中的每一行代表一部电影或者电视节目，包含电影导演、演员等信息。如果某人与该行电影有关则用 1 表示，否则用 0 表示。

IMDB 有正规的电影评分系统，用户可以根据喜好对电影评分，从而可以将电影分为正面情感和负面情感两类。该数据集包含 2.5 万条电影评论，每一条都被赋予正面或负面标签，且每一条评论被处理成一个序列。

本实验所用的数据集，是 Bengio 团队将原始数据集封装后得到的。包括训练集、测试集、验证集。每个数据集的序列数据和对应标签以数组的形式封装在一起。例如：train set(0) 代表训练集的序列数据，train set(1)代表训练集的类标，将二者共同用来训练网络。

本实验所用训练集数据量为 1998 条，测试集数据为 250 000 条(实际用于网络的数量可自定义)，验证集数据为 105 条。其中训练数据用来训练网络的参数，验证集用来精调网络，测试集用来检验网络的性能。网络的类标为 0 或 1 标签，其中 1 代表正面情感，0 代表负面情感，一个数据对应着一个类标。序列数据并没有固定长度，对于 RNN 来说，恰好不受输入长度的限制，另外情感语义的分析需要参照上下文，可以利用 RNN 的记忆功能，而 LSTM 在传统 RNN 基础上做了改进，可以更好地进行记忆和遗忘。所以使用 LSTM 进行情感分类是高效简便的方式。

2. 模型介绍

本实验运用 Theano 平台 LSTM 的官方教程提供的代码，网络参数设置如下：LSTM 隐层单元数：128，同时也是一个词转为一个向量的维数；Batch_size：训练过程为 16，验证过程为 64；词典大小：10000，给每个词赋予一个 ID，ID 号为 1～10000，词的 ID 代表在 embedding 矩阵中的第几行；序列长度：100，超出序列长度的部分抛弃。

3. 优化方法

权值初始化：使用[0,1]随机数生成矩阵，再对随机矩阵进行 SVD 分解；网络优化方法：代码提供了 SGD、Adadelta 和 Rmsprop 三种方法，本实验采用自适应学习率调整(Adadelta)；损失函数：softmax 函数，加 L2 范数正则项。

4. 实验结果

网络设置显示频率：10，保存频率：1110；验证频率：370。当运行到显示频率，就输出对应的步数，更新代数，误差值；运行到保存频率，就将当前参数保存；运行到验证频率，就计算误差，并更新最优值。具体显示如图 11.38 所示。

```
Seen 1998 samples
Epoch  23 Update  2880 Cost  0.00244462345929
Epoch  23 Update  2890 Cost  0.00155562072142
Epoch  23 Update  2900 Cost  0.00588232023833
Epoch  23 Update  2910 Cost  0.00688468451577
Epoch  23 Update  2920 Cost  0.00774977262055
Epoch  23 Update  2930 Cost  0.00788612011513
Epoch  23 Update  2940 Cost  0.0196560150077
Epoch  23 Update  2950 Cost  0.00349733390678
Epoch  23 Update  2960 Cost  0.00487144610723
```

图 11.38　Theano LSTM 运行过程显示图

网络的训练准确率：99%；测试准确率：80.2%；验证准确率：86.7%。运行结果如图 11.39 所示。

```
Seen 1998 samples
Epoch  76 Update  9510 Cost  2.55437011678e-06
Epoch  76 Update  9520 Cost  4.23855318374e-05
Epoch  76 Update  9530 Cost  1.46567246382e-05
Epoch  76 Update  9540 Cost  8.87037114227e-06
Epoch  76 Update  9550 Cost  7.3983283929e-07
Epoch  76 Update  9560 Cost  3.35468880068e-06
Epoch  76 Update  9570 Cost  6.07715280172e-06
Epoch  76 Update  9580 Cost  2.41964258536e-05
Epoch  76 Update  9590 Cost  5.99396759225e-06
Epoch  76 Update  9600 Cost  8.59450039217e-06
Epoch  76 Update  9610 Cost  2.24269006787e-06
Epoch  76 Update  9620 Cost  5.09386337933e-05
Train  0.0 Valid  0.152380952381 Test  0.222
Early Stop!
Seen 1920 samples
Train  0.0105105105105 Valid  0.133333333333 Test  0.198
The code run for 77 epochs, with 58.461746 sec/epochs
Training took 4501.6s
```

图 11.39　Theano LSTM 运行结果图

参考文献

[11.1] Jia，Yangqing，Shelhamer，et al. Caffe：Convolutional Architecture for Fast Feature Embedding [J]. Eprint Arxiv，2014：675-678.

[11.2] Krizhevsky A，Sutskever I，Hinton G E. ImageNet classification with deep convolutional neural networks[C]. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012：1097-1105.

[11.3] Yuan Z W，Zhang J. Feature extraction and image retrieval based on AlexNet[C]. Eighth International Conference on Digital Image Processing. 2016：100330E.

[11.4] Korabelnikov A N，Kolsanov A V，? Haplygin S S，et al. Liver tumor segmentation CT data based on Alexnet-like convolution neural nets [C]. International Conference Information Technology and Nanotechnology. 2016：348-356.

[11.5] Abadi M，Agarwal A，Barham P，et al. TensorFlow：Large-Scale Machine Learning on Heterogeneous Distributed Systems[J]. 2016.

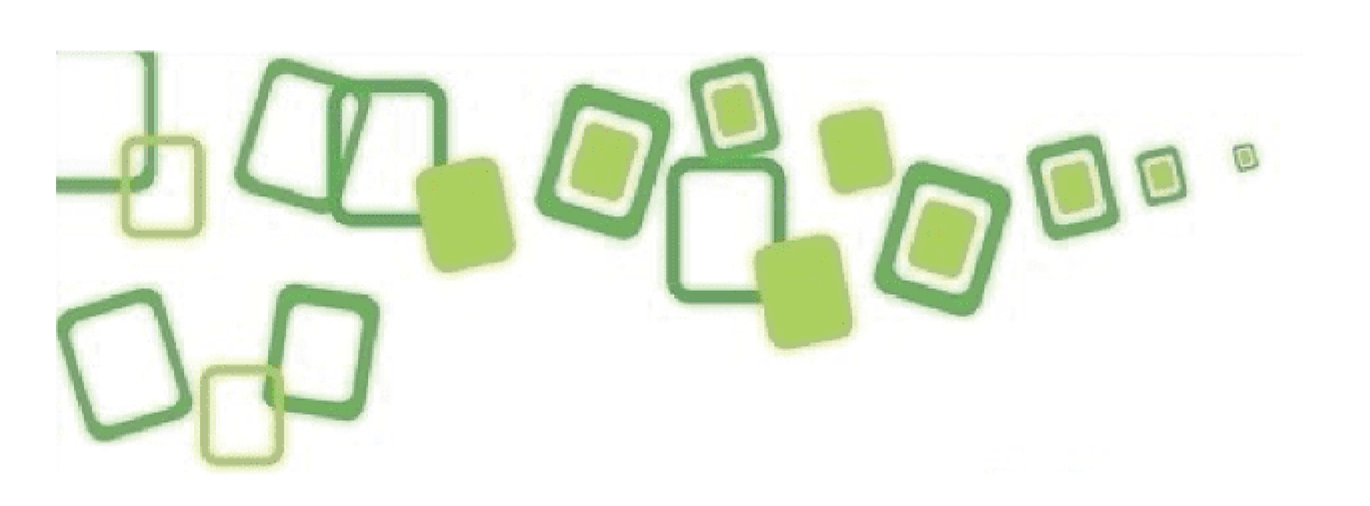
[11.6] Abadi M，Barham P，Chen J，et al. TensorFlow：A system for large-scale machine learning [J]. 2016.

[11.7] Goodfellow I，Pougetabadie J，Mirza M，et al. Generative Adversarial Nets[J]. Advances in Neural Information Processing Systems，2014：2672-2680.

[11.8] Radford A，Metz L，Chintala S. Unsupervised Representation Learning with Deep Convolutional

Generative Adversarial Networks[J]. Computer Science, 2015.
[11.9] Creswell A, Bharath A A. Adversarial Training for Sketch Retrieval[J]. 2016.
[11.10] Chen T, Li M, Li Y, et al. MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems[J]. Statistics, 2015.
[11.11] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science, 2014.
[11.12] Chatfield K, Simonyan K, Vedaldi A, et al. Return of the Devil in the Details: Delving Deep into Convolutional Nets[J]. Computer Science, 2014.
[11.13] Deng J, Dong W, Socher R, et al. ImageNet: A large-scale hierarchical image database[J]. 2009: 248-255.
[11.14] Collobert R, Kavukcuoglu K, Farabet C. Torch7: A Matlab-like Environment for Machine Learning [C]. BigLearn, NIPS Workshop. 2011.
[11.15] Ierusalimschy R, Figueiredo L H D, Filho W C. Lua—An Extensible Extension Language[J]. Software Practice & Experience, 1996, 26(6): 635-652.
[11.16] Courbariaux M, Hubara I, Soudry D, et al. Binarized Neural Networks: Training Deep Neural Networks with Weights and Activations Constrained to +1 or −1[J]. 2016.
[11.17] Wang S, Ren D, Chen L, et al. On Study of the Binarized Deep Neural Network for Image Classification[J]. 2016.
[11.18] Bergstra J, Breuleux O, Bastien F, et al. Theano: A CPU and GPU math compiler in Python [J]. 2010: 3-10.
[11.19] Team T D, Alrfou R, Alain G, et al. Theano: A Python framework for fast computation of mathematical expressions[J]. 2016.
[11.20] Graves A. Supervised Sequence Labelling with Recurrent Neural Networks[M]. Springer Berlin Heidelberg, 2012.
[11.21] Graves A, Mohamed A R, Hinton G. Speech recognition with deep recurrent neural networks [J]. 2013, 38(2003): 6645-6649.
[11.22] Sundermeyer M, Schlüter R, Ney H. LSTM Neural Networks for Language Modeling[C]. Interspeech. 2012: 601-608.

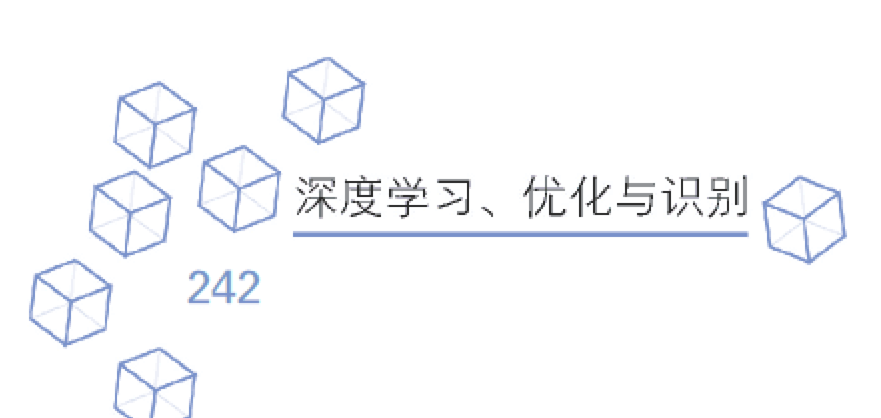
第12章



基于深度神经网络的SAR/PolSAR影像地物分类

CHAPTER 12

SAR/PolSAR影像分类——深度神经网络



12.1　数据集及研究目的

12.1.1　数据集特性分析

合成孔径雷达(Synthetic Aperture Radar,SAR)首次使用于20世纪中期。雷达的分辨率与孔径成正比,SAR的基本原理是让雷达在一条直线上运动,将雷达在这条直线上收到的数据合成起来,就相当于一个直径等于直线距离的大孔径雷达。即使目标的能见度很差,SAR也可以得到清晰的图像。它的特点是所获得的图像分辨率高,可以7×24小时工作,能发现被遮挡的地物。这些特点使其不仅在农、林、自然灾害等民用领域具有广泛的应用前景,在军事领域更具发展潜力。

1. 极化特性描述

移动天线上产生的电力场中的带电粒子就可以产生电磁波,它的传播不需要任何媒质,其具有极化特性。

场的复振幅在任意方向上可以用如下公式描述:

$$E(r)=\hat{E}_0\cdot \mathrm{e}^{i\cdot k\cdot e_{\mathrm{p}}\cdot r} \tag{12.1}$$

$$H(r)=\hat{H}_0\cdot \mathrm{e}^{i\cdot k\cdot e_{\mathrm{p}}\cdot r},\quad H(r)=\hat{H}_0\cdot \mathrm{e}^{i\cdot k\cdot e_{\mathrm{p}}\cdot r} \tag{12.2}$$

其中 $\boldsymbol{e}_{\mathrm{p}}$ 是平行于传播方向的单位矢量。从式(12.1)和式(12.2)可以得出,电场矢量已知,可求得磁场矢量,磁场矢量已知,可求得电场矢量,只讨论电场矢量即可。电场矢量可以表示为:

$$E(r,t)=\mathrm{Re}(E_0\cdot \mathrm{e}^{i(w\cdot t+k\cdot r)}) \tag{12.3}$$

将其分解为 x 分量和 y 分量:

$$E(z)=E_x(z)\cdot e_x+E_y(z)\cdot e_y \tag{12.4}$$

令:

$$E_x(z,t)=E_{x_0}\cdot\cos(w\cdot t-k\cdot z) \tag{12.5}$$

$$E_y(z,t)=E_{y_0}\cdot\cos(w\cdot t-k\cdot z-\varphi_0) \tag{12.6}$$

如果相位差 φ_0 为0,合成电场强度为:

$$E(z,t)=\sqrt{E_{x_0}^2+E_{y_0}^2}\cdot\cos(w\cdot t-k\cdot z) \tag{12.7}$$

此时的电场为线极化,如果 φ_0 为90,电场为圆极化,一般情况下,φ_0 既不等于0,又不等于90,电场为椭圆极化。

2. 描述矩阵

1984年,George Sinclair描述了极化散射矩阵这一理论,其用一个矩阵元素为复数的二维矩阵来描述。电磁波发射至散射体这一过程可表示为:

$$E^{tt} = E_H^{tt} \cdot e_H + E_V^{tt} \cdot e_V \tag{12.8}$$

被散射体弹回后，整个发射弹回的流程如果用[**S**]描述，则接收场是

$$E^{rr} = [\boldsymbol{S}] \cdot E^{tt} = \begin{bmatrix} E_H^{rr} \\ E_V^{rr} \end{bmatrix} = \frac{e^{i \cdot k_0 \cdot r}}{r} \cdot \begin{bmatrix} S_{HH} & S_{HV} \\ S_{VH} & S_{VV} \end{bmatrix} \cdot \begin{bmatrix} E_H^{tt} \\ E_V^{tt} \end{bmatrix} \tag{12.9}$$

其中，上标 tt 代表入射电磁波，上标 rr 表示散射电磁波，r 代表地物离发射电磁波的地方有多远，K_0 为电磁波的波数。[**S**]为极化散射矩阵。S_{HH}、S_{VV} 称为同极化分量，S_{HV}、S_{VH} 称为交叉极化分量。极化协方差矩阵[**C**]为

$$[\boldsymbol{C}] = \begin{bmatrix} \langle S_{HH}, S_{HH}^* \rangle & \sqrt{2}\langle S_{HH}, S_{HV}^* \rangle & \langle S_{HH}, S_{VV}^* \rangle \\ \sqrt{2}\langle S_{HV}, S_{HH}^* \rangle & 2\langle S_{HV}, S_{HV}^* \rangle & \sqrt{2}\langle S_{HV}, S_{VV}^* \rangle \\ \langle S_{VV}, S_{HH}^* \rangle & \sqrt{2}\langle S_{VV}, S_{HV}^* \rangle & \langle S_{VV}, S_{VV}^* \rangle \end{bmatrix} \tag{12.10}$$

其中〈·，·〉表示空间统计平均。极化相干矩阵[**T**]为

$$[\boldsymbol{T}] = \begin{bmatrix} \langle (S_{HH}+S_{VV}), (S_{HH}+S_{VV})^* \rangle & \langle (S_{HH}+S_{VV}), (S_{HH}-S_{VV})^* \rangle & \langle 2(S_{HH}+S_{VV}), S_{HV}^* \rangle \\ \langle (S_{HH}-S_{VV}), (S_{HH}+S_{VV})^* \rangle & \langle (S_{HH}-S_{VV}), (S_{HH}-S_{VV})^* \rangle & \langle 2(S_{HH}-S_{VV}), S_{HV}^* \rangle \\ \langle 2S_{HV}, (S_{HH}+S_{VV})^* \rangle & \langle 2S_{HV}, (S_{HH}-S_{VV})^* \rangle & \langle 4S_{HV}, S_{HV}^* \rangle \end{bmatrix} \tag{12.11}$$

其中 $\boldsymbol{A}=S_{HH}+S_{VV}$，$\boldsymbol{B}=S_{HH}-S_{VV}$，$\boldsymbol{C}=2S_x$。

3. 极化 SAR 特征提取方法

1）测量数据简单变换和组合的特征

基于散射矩阵[**S**]的特征主要有：S_{HH}、S_{VV}、S_{HV}、$S_{HH}+S_{VV}$、$S_{HH}-S_{VV}$ 等。基于功率测量的特征由矩阵[**T**]和[**C**]中的元素构成。

2）目标分解的特征

目前，目标分解的方法主要有两个：第一个主要面向矩阵[**S**]，是相干目标分解；第二个针对[**C**]和[**T**]，称为非相干目标分解。

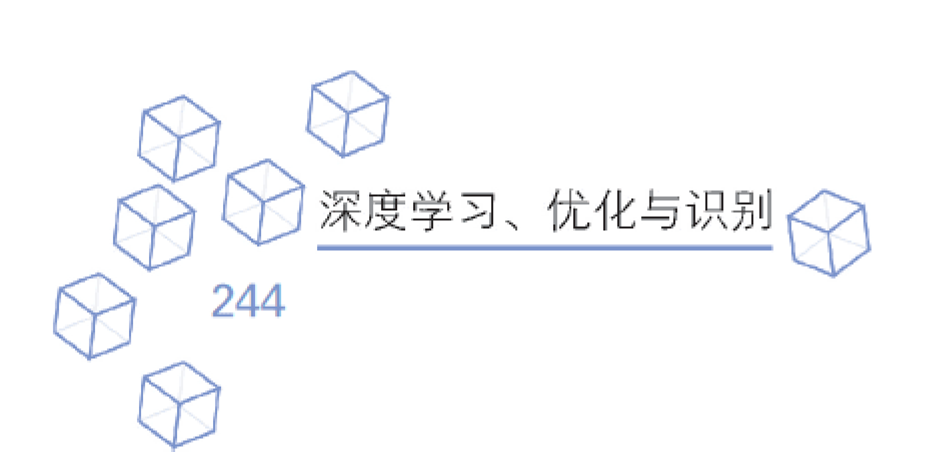
第一个方法是将[**S**]分解成各种地物目标的[**S**]相加，它们都代表了现实中某种确定的物理散射机制，过程如下：

$$[\boldsymbol{S}] = \sum_{i=1}^{k} c_i [\boldsymbol{S}]_i \tag{12.12}$$

其中，$[\boldsymbol{S}]_i$ 代表各种确定的地物目标的散射矩阵，c_i 代表每种已知地物目标所占的比例。

相干目标分解主要包括 Pauli、Krogager、Cameron、SSCM 分解等。因为相干分解是针对极化散射矩阵的，分解是针对单个像素点的，可以尽可能地维持极化 SAR 影像的原有特性。但是相干分解并没有考虑极化 SAR 图像的相干斑噪声，也没有考虑一些复杂地物目标的统计起伏性，所以存在一定的缺陷。所以目前使用最多的为上文所述的第二种方法——非相干目标分解方法。

极化非相干目标分解的主要思想是把[**C**]或者[**T**]表示为简单标准的二阶描述子的组合，表示为：



$$\langle[\boldsymbol{C}]\rangle=\sum_{i=1}^{k}p_i[\boldsymbol{C}]_i \tag{12.13}$$

$$\langle[\boldsymbol{T}]\rangle=\sum_{i=1}^{k}q_i[\boldsymbol{T}]_i \tag{12.14}$$

其中 p_i，q_i 表示系数，$[\boldsymbol{C}]_i$、$[\boldsymbol{T}]_i$ 表示分量响应。非相干分解的方法包括 Cloude 分解、Freeman-Durden 分解、Huynen 分解、Yamaguchi 分解等。其中最经典的方法为 Cloude 分解和 Freeman-Durden 分解。

1986 年，Cloude 等人提出了 Cloude 分解，这种分解方法不但可以解释物理散射机制，还具有正交散射机制的优点。$[\boldsymbol{T}]$可以表示为三个互相独立的相干矩阵的和。

$$[\boldsymbol{T}]=\sum_{i=1}^{3}\lambda_i[\boldsymbol{T}_i]=\lambda_1 e_1\cdot e_1^*+\lambda_2 e_2\cdot e_2^*+\lambda_3 e_3\cdot e_3^* \tag{12.15}$$

其中 e_i 是特征向量，λ_i 是特征值。

Cloude、Pottier 定义散射熵 H 如下式所示，其用来描述煤质散射的随机性。

$$H=\sum_{i=1}^{3}-(P_i\cdot\log_3 P_i) \tag{12.16}$$

其中 $P_i=\dfrac{\lambda_i}{\sum_j\lambda_j}$。

散射角 α 大小为多少要根据目标的物理散射机制来确定，$\alpha=0$ 表示奇次散射(或称为表面散射)，$\alpha=45$ 表示偶极子散射(或称为体散射)，$\alpha=90$ 表示偶次散射(或二面角散射)，这里散射角是一种平均散射机制，故记为 $\bar{\alpha}$。

$$\bar{\alpha}=P_1\cdot\alpha_1+P_2\cdot\alpha_2+P_3\cdot\alpha_3 \tag{12.17}$$

H 和 $\bar{\alpha}$ 刻画了煤质散射特性，它们可以将整个空间分割成 8 个部分，每个部分对应着某种散射机制。

1998 年，Freeman、Durden 提出 Freeman-Durden 分解方法，给$[\boldsymbol{C}]$或$[\boldsymbol{T}]$建立了如图 12.1 所示的三种散射机制的模型。

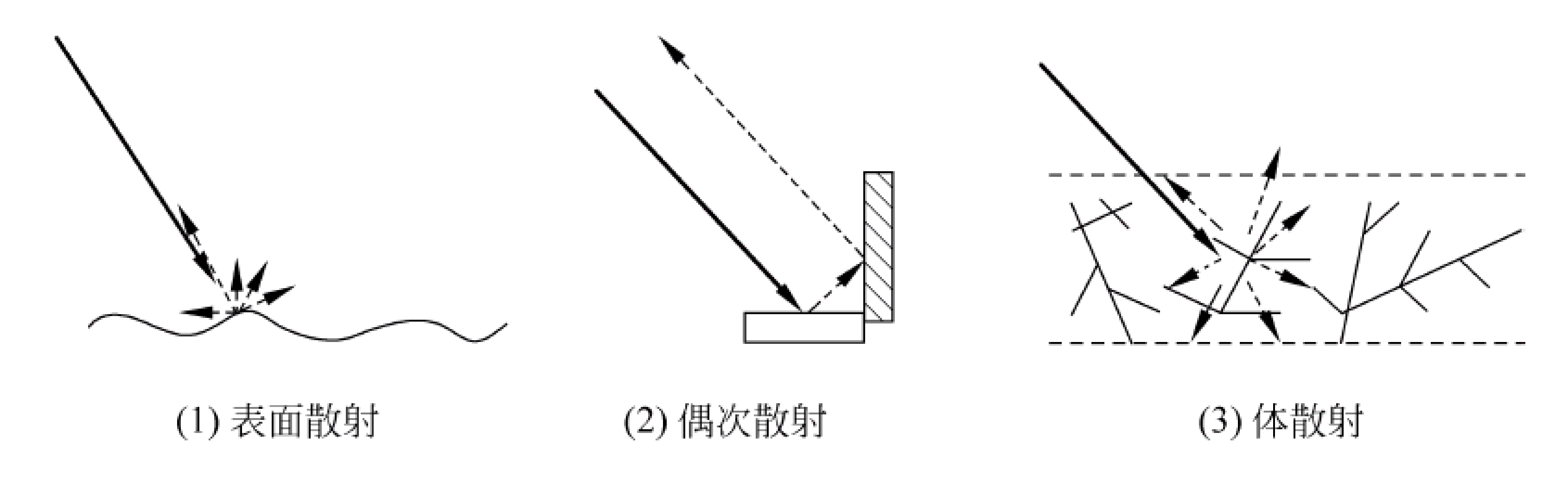


图 12.1　三种基本散射机制

表面散射的$[\boldsymbol{C}]$如下式所示：

$$[\boldsymbol{C}]_s=f_s\cdot\begin{bmatrix}|\beta|^2 & 0 & \beta\\ 0 & 0 & 0\\ \beta^* & 0 & 1\end{bmatrix} \tag{12.18}$$

其中 $\beta=\dfrac{R_H}{R_V}$，f_s 表示布拉格表面散射分量权值系数。

偶次散射能够用角反射器模拟，其中二面角散射分量的[$\boldsymbol{C}$]如下式所示：

$$[\boldsymbol{C}]_d = f_d \cdot \begin{bmatrix} |\alpha|^2 & 0 & \alpha \\ 0 & 0 & 0 \\ \alpha^* & 0 & 1 \end{bmatrix} \tag{12.19}$$

其中 $\alpha=\dfrac{R_{gh}R_{vh}}{R_{gv}R_{vv}}$，$R_{gh}$表示地表的水平菲涅尔系数，$R_{gv}$表示地表的垂直菲涅尔系数，$R_{vh}$、$R_{vv}$表示竖直墙体的菲涅尔系数，$f_d$ 表示二面角散射分量的权值系数。

体散射的[$\boldsymbol{C}$]如下式所示：

$$[\boldsymbol{C}]_v = f_v \cdot \begin{bmatrix} 1 & 0 & 1/3 \\ 0 & 2/3 & 0 \\ 1/3 & 0 & 1 \end{bmatrix} \tag{12.20}$$

式(12.20)中，f_v 表示体散射分量权值系数。Freeman-Durden 分解是把[$\boldsymbol{C}$]表示成上述三个值之和：

$$[\boldsymbol{C}] = [\boldsymbol{C}]_v + [\boldsymbol{C}]_d + [\boldsymbol{C}]_s \tag{12.21}$$

为了求得式(12.21)，假设散射体是对称的，而且是互易的，总的后向散射模型表示如下：

$$\begin{cases} \langle |S_{HH}|^2 \rangle = f_s \cdot |\beta|^2 + f_d \cdot |\alpha|^2 + f_v \\ \langle |S_{VV}|^2 \rangle = f_s + f_d + f_v \\ \langle |S_{HH}S_{VV}^*|^2 \rangle = f_s \cdot \beta + f_d \cdot \alpha + f_v/3 \\ \langle |S_{HV}|^2 \rangle = f_v/3 \end{cases} \tag{12.22}$$

可得到：

$$\begin{cases} P_s = f_s \cdot (1 + |\beta|^2) \\ P_d = f_d \cdot (1 + |\alpha|^2) \\ P_v = 8f_v/3 \\ p = P_s + P_d + P_v = |s_{hh}|^2 + 2|s_{hv}|^2 + |s_{vv}|^2 \end{cases} \tag{12.23}$$

P_s 是表面散射分量的散射能量，P_d 是二面角散射分量的散射能量，P_v 是偶极散射分量的散射能量。

12.1.2　基本数据集

1. 常用 SAR 数据集

图 12.2 是一幅 Radarsat 卫星拍摄的水域 SAR 图像，大小为 254×255，包含河流与陆地两类地物。

图 12.3 是一幅位于美国新墨西哥州 Albuquerque 地区附近的 RioGrande river 区域，

分辨率为 1m，大小为 510×510，Ku 波段，中间为一个机场跑道的三类简单地物 SAR 图像。

图 12.4 为一幅美国加州 china lake 机场地区的 Ku 波段 SAR 图像，分辨率为 3m，大小为 256×256，主要包含三类地物：城市、跑道和农田。

图 12.5 为一幅位于美国新墨西哥州 Albuquerque 地区附近的 RioGrande river 流域输油管的 Ku 波段 SAR 图像，分辨率为 1m，，大小为 256×257，主要包含三类地物：灌木丛、草地和河流。

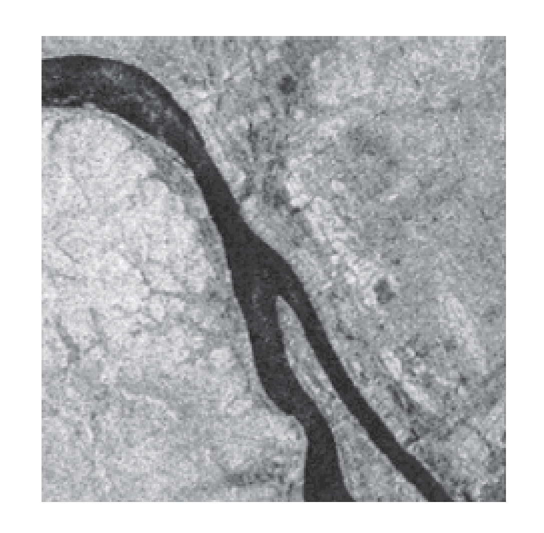


图 12.2　水域 SAR 图像

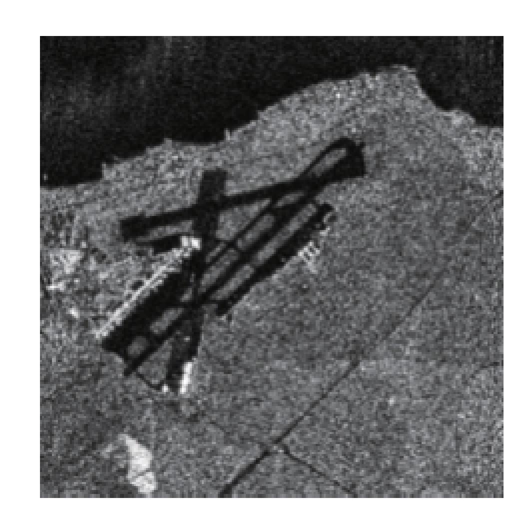


图 12.3　RioGrande river 区域

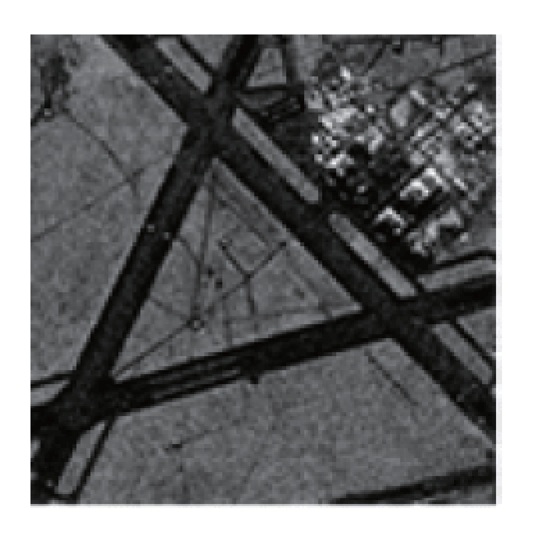


图 12.4　china lake 机场地区

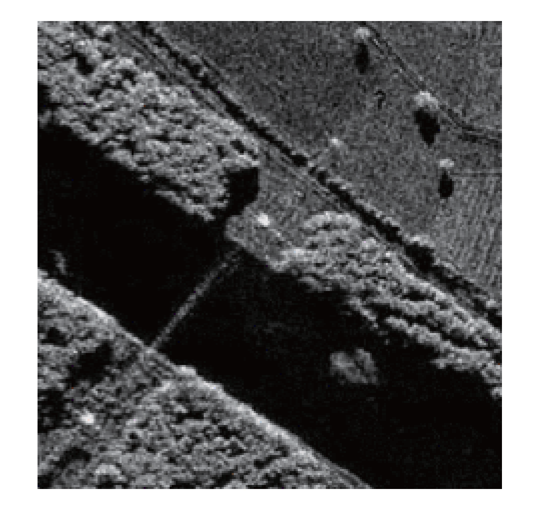


图 12.5　RioGrande river 流域输油管

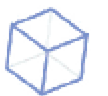
2. 常用 PolSAR 数据集

1) Flevoland 农田数据

Flevoland 数据是 1989 年，NASA/JPL 实验室 AIRSAR 系统在 L 波段获得关于 Flevoland 的 Netherlands 的农田全极化数据（官方链接：http://earth. eo. esa. int/polsarpro/default. html. ），该数据在极化 SAR 分类算法中被广泛使用，而且在分类前未对数据进行滤波处理。该数据大小为 750×1024，其 PauliRGB 图像如图 12.6(a)所示，图 12.6(b)是此图的 Groundtruth，图 12.6(c)是此图的颜色编码。

2) Germany 数据

数据来自 Germany 的 Oberpfaffenhofen 地区，它是机载 ESAR 获得的 L 波段的多视数据。本节选取的图像大小为 1300×1200，其中主要包括三类主要地物：建筑区、草木区和空旷区。该数据的 PoliRGB 图像如图 12.7(a)所示，图 12.7(b)是此图的 Groundtruth，图 12.7(c)是此图的颜色编码。



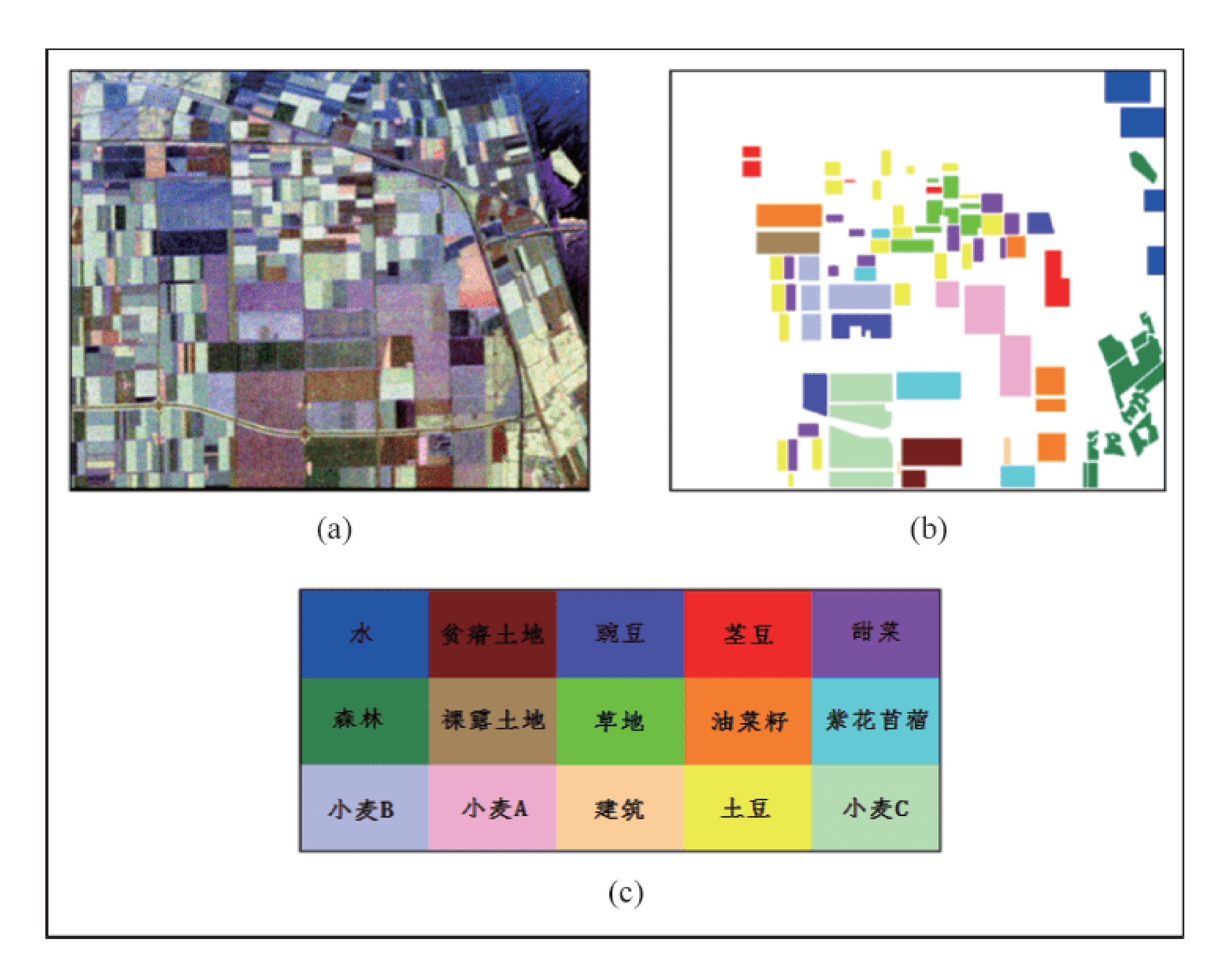


图 12.6 Flevoland 农田数据

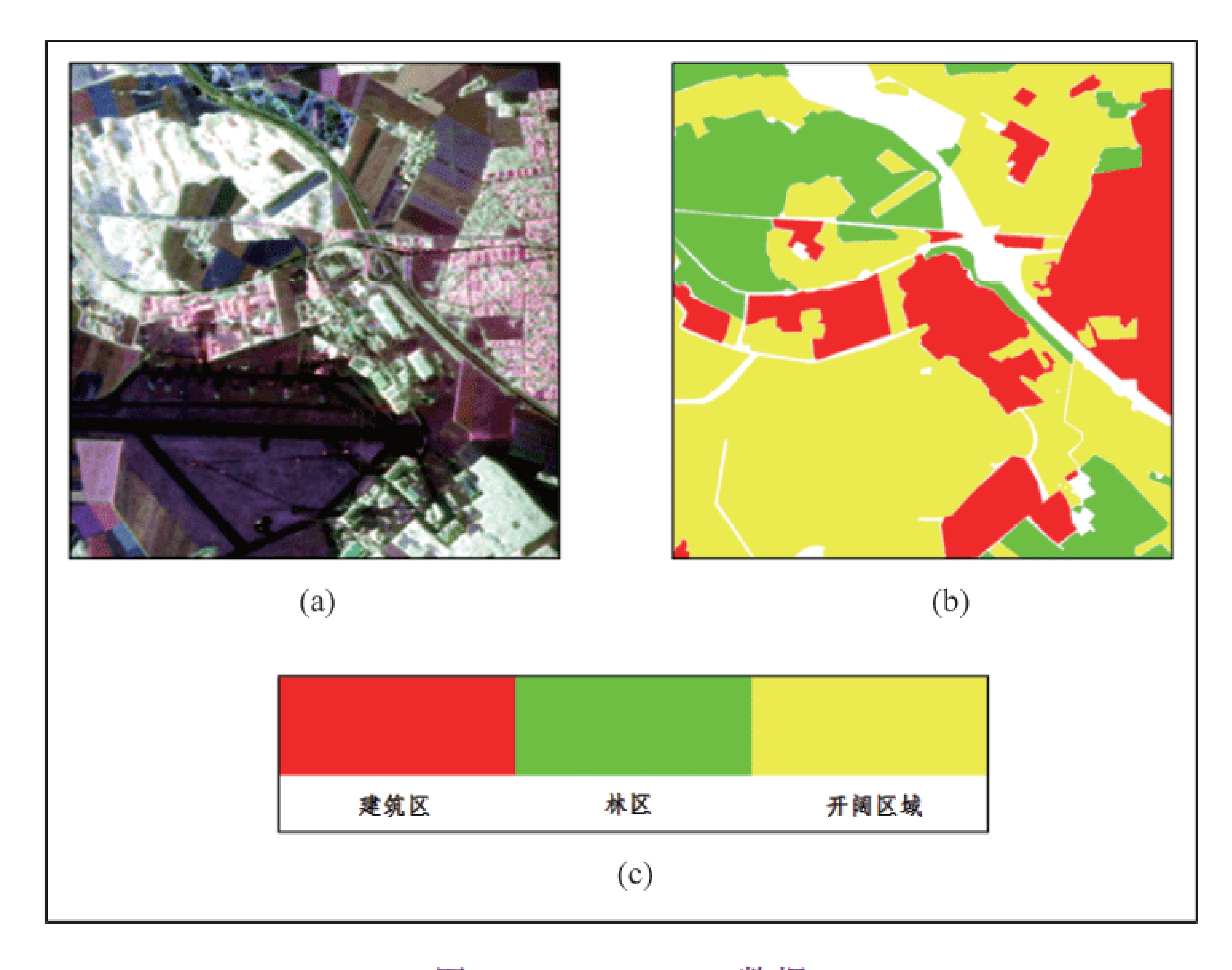


图 12.7 Germany 数据

3）San Fracisco 数据

数据来自 San Fracisco，美国，拍摄时间为 2008 年 4 月，分辨率为 10×5m，本文选取的图像大小为 1800×1380，其中主要包括五类不同的区域：海洋、植被、发达区域、人口密集度



集中的城区和人口密集度低的城区。该数据的 Pauli RGB 图像如图 12.8(a)所示,图 12.8(b)是此图的 Ground truth,图 12.8(c)是此图的颜色编码。

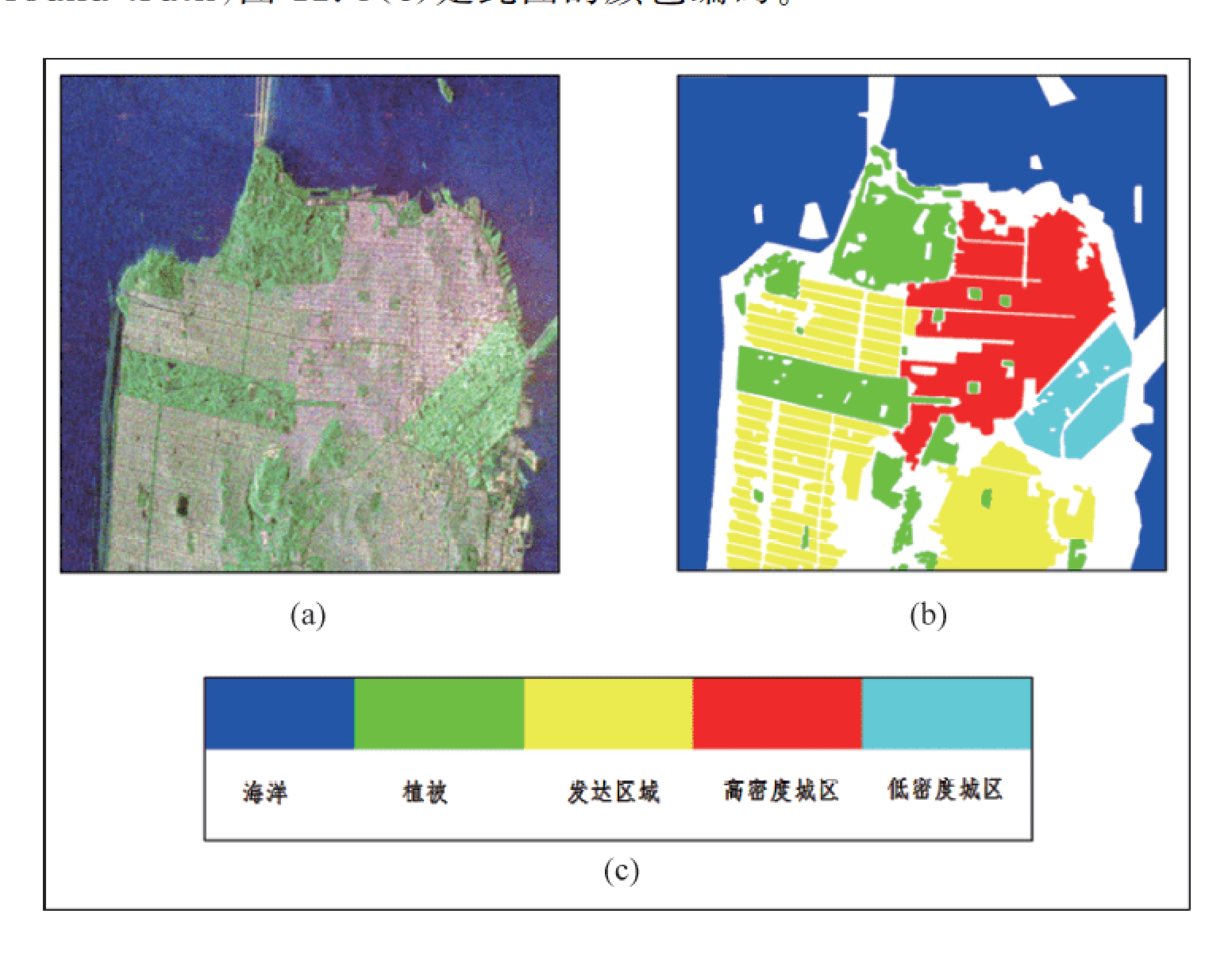


图 12.8 San Francisco 数据

12.1.3 研究目的

SAR 图像地物分类,也称为 SAR 土地覆盖(Land-Cover),是 SAR 图像的一个重要应用。SAR 地物分类的主要任务是利用 SAR 图像自身特性与其他手段来确定对应区域所属的地物类别。但是由于成像方式特殊,与光学图像相比较,SAR 图像可读性较差,场景理解比较困难,这就导致了 SAR 图像地物分类异常困难。SAR 地物分类使用的特征描述主要是以下三个方面。

1. 电磁散射特征

由于 SAR 成像系统工作在微波波段,因此具有不同材料及不同结构的地物对不同波段的电磁波具有不同的后向散射特性。基于此,通过建立雷达后向散射系数与不同类型地物之间的关系可以对被观测场景中的不同地物类型进行有效区分。

例如:通过对城区图像中由不同建筑结构引起的单边反射、双边反射及三边散射特性进行分析并用于城市监测和分类。通过对 ERS-1、JERS-1 及 SIR-C 卫星的平均散射系数与中央亚马逊河流的热带雨林的覆盖密度之间关系的研究发现,具有较长波长的电磁波,如 L 波段,对植被密度的变化较为敏感,而具有较短波长的电磁波,如 C 波段,对植被与干旱的裸地之间的区分能力较为有限。因此位于 L 波段的日本 ALOS PALSAR 系列卫星所获取

的 SAR 图像被广泛应用于全球森林覆盖监控、湿地生态系统监控及农作物分类及长势监测等领域。

L. Hess 等学者通过对 1994 年 4 月到 10 月巴西玛瑙斯市附近的亚马逊河流域受淹滩区及各种植被的 SIR-C 多频 SAR 数据进行分类(水域、牧场、水生植物、非淹森林及水淹森林)的研究结果表明,C 波段及 L 波段的电磁波能很好区分这几类不同的地物。同时研究结果也表明 HH 极化方式对水淹植被的区分能力较好,而 L 波段的交叉极化数据对木质及非木质植被的区分能力较强。基于 C 波段 RADARSAT-Ⅱ卫星的观测数据,Y. Inoue 等学者对水稻长势的研究结果表明 C 波段的雷达后向散射系数与光合有效辐射比(frAction of Photosynthetically active Radiation,fAPRA)具有较高的线性相关性,可被用于监测水稻的生长状态。

众多的研究结果表明,通过对多种不同地物的后向散射机制进行研究能有效对各种不同地物类型进行分类,特别是 L 波段的电磁波对不同的植被具有良好的区分能力。通常情况下,SAR 图像的后向散射特征对电磁波固定散射特性与地物类别之间关系模型的依赖度较高。实际应用中,如果能对依赖模型获取准确的先验知识,则通过后向散射特性可以准确地判别各种不同地物类别而无须额外的“监督”信息。然而对后向散射信息的准确提取过分依赖于 SAR 图像的准确校正、详细且精确的成像参数以及大量全面数据分析的专业知识。因此其难以广泛应用于不同 SAR 成像系统及不同成像参数下所获取的 SAR 图像。与此同时,大量全面的数据分析需要耗费专业的人力资源去读取、检索及判别 SAR 图像的内容,这无疑又增加了 SAR 系统获取图像信息的时间及人力成本。与此同时,随着 SAR 图像空间分辨率的不断提升,中低分辨图像所得到的散射模型是否依旧适用于高分辨 SAR 图像仍是一个值得商榷的问题。

2. 强度或幅度的统计特征

当 SAR 图像的空间分辨率较低时,雷达的回波信号中包含了一个分辨单元中的所有基本散射体的回波。受分辨单元内散射体多样性(不同形状、不同材质、不同距离等)的影响,所接收的雷达回波在相位上会产生严重的不相干现象。因此,通过对雷达的连续脉冲回波进行相干处理之后,各个分辨单元内散射体的差异导致所形成的 SAR 图像中含有大量颗粒状的变化模式,这种现象通常被称为相干斑。在 SAR 图像中,这种相干斑现象不是噪声,其中包含着 SAR 传感器及被观测地物的特定信息,因此,对各种不同的地物类型的相干斑进行统计建模也是对 SAR 特征提取重要的研究内容之一。

近年来,大量的统计模型被用于对 SAR 图像中不同地物的相干斑统计模型进行建模,并成功用于 SAR 图像地物分类及相干斑抑制,主要包括指数、Gamma、Weibull、对数正态(log-Normal)及 K 分布等。具体来说,在 SAR 图像处理领域中较为常见的概率同分布模型及其使用场景如表 12.1 所示。基于这些统计分布,对 SAR 图像的分类问题可视为一个参数估计问题。通过一个精确的参数估计过程,如经典的最大似然估计,矩估计或者对数矩估计等,各种不同的地物可以通过对应分布的参数进行描述。

表 12.1　部分常见的对 SAR 图像进行统计建模的概率分布

统计模型	概率表示	应用场景
指数	$p(x)=\frac{1}{\sigma}\exp\left(-\frac{x}{\sigma}\right)$	匀质场景，单视 SAR 强度数据
Gamma	$p(x)=\frac{1}{\sigma\cdot\Gamma(\sigma)}\left(\frac{x}{\sigma}\right)\exp\left(-\frac{x}{\sigma}\right)$	多视 SAR 强度数据
Weibull	$p(x)=\frac{\beta}{\sigma}\left(\frac{x}{\sigma}\right)^{\beta}\exp\left(-\left(\frac{x}{\sigma}\right)^{\beta}\right)$	低异质性的 SAR 强度数据
Rayleigh	$p(x)=\frac{x}{b^2}\exp\left(-\frac{x^2}{2b^2}\right)$	单视匀质 SAR 幅值数据
对数正态	$p(x)=\frac{1}{\sqrt{2\pi\sigma x}}\exp\left(-\frac{(\log x-m)^2}{2b^2}\right)$	非常异质性表面，如城区 SAR 数据
Nakagami-Rice	$p(x)=\frac{1}{R}\exp\left(-\frac{x+A_0^2}{R}\right)I_0\left[2\frac{\sqrt{A_0^2x}}{R}\right]$	存在单一强散射体的匀质散射区域
K	$p(x)=\frac{1}{\Gamma(L)\Gamma(M)}\cdot\frac{2LM}{\mu}\cdot\left(\frac{LM}{\mu}\cdot x\right)^{\frac{L+M}{2}-1}\cdot K_{M-L}$	中度异质 SAR 幅度或强度数据
Fisher	$P(x)=\frac{\Gamma(L+M)}{\Gamma(L)\Gamma(M)}\frac{L}{M\mu}\frac{\left(\frac{Lx}{M\mu}\right)^{L-1}}{\left(1+\left(\frac{Lx}{M\mu}\right)\right)^{L+M}}$	各类 SAR 图像幅度或强度数据
G^0	$p(x)=\frac{n^n\Gamma(n-\alpha)\gamma^{-\alpha}x^{n-1}}{\Gamma(n)\Gamma(-\alpha)(\gamma-nx)^{n-\alpha}}$	各类 SAR 图像幅度或强度数据

3. SAR 图像特征提取及学习

与后向散射特性及统计模型不同的是，SAR 图像特征提取与学习过程是对人类的视觉机理过程进行模拟，借助计算机对 SAR 图像进行自动分析和理解，以提取相关信息对各种 SAR 地物类型进行描述。由于与传统光学图像的成像机理不同，对 SAR 图像的特征提取与学习更为困难。

在各种视觉特征描述中，通过对 SAR 图像上相隔一定距离的两个像素灰度值之间的空间关系进行统计分析可得到灰度共生矩阵(Gray Level Co-occurrence Matrix，GLCM)，并由此计算各种纹理描述子对 SAR 图像中不同地物纹理进行描述。较为常用的纹理描述统计量有能量、角二阶矩、对比度/反差、相关性、匀质性、差分矩、熵、和平均及差熵等。基于这些纹理描述子，L. Soh 等学者对海冰分类的研究结果表明在灰度共生矩阵的计算中，像素之间的间隔要比方向因子对海冰分类性能影响更为明显。由于对灰度共生矩阵的计算代价是 SAR 图像像素距离和角度的矩阵函数，因此完整的灰度共生矩阵计算是费时费力的。此外对 SAR 图像的分类任务而言，如何选取合适的统计方向、统计距离等参数以充分有效地提

取信息对 SAR 图像进行描述在实际中也是一大挑战。

与由灰度共生矩阵所计算得到的纹理描述子不同的是：图像的局部性特征提取也是近年来 SAR 特征提取的重要方面。各种多尺度多方向滤波器组，小波及二代小波变换等也被广泛应用于 SAR 图像特征描述。其中，Gabor 滤波器的尺度和方向表示更接近人类视觉系统对频率和方向的表示，因而被广泛应用于描述 SAR 图像中地物的结构信息。基于离散小波变换，M. Tello 等学者通过利用变换后的各个变换子带之间的空间关系和船舶及海洋的统计行为对 SAR 图像中的舰船目标进行鲁棒有效的监测。除了小波变换之外，大量二代小波变换，如 bandlet 变换、contourlet 变换、curvelet 变换及 ridgelet 变换等也被用于获取 SAR 图像在各种尺度及方向上的特征表示，并用于 SAR 图像的分类及分割操作。相对于 SAR 图像的特征信息提取而言，众多的小波及二代小波变换等被广泛用于 SAR 图像去斑及压缩。近年来，F. Dellinger 等研究者通过结合 SAR 图像的统计分布特性，将计算机视觉领域被广泛使用的 SIFT(Scale Invariant Feature Transform)特征扩展到 SAR 图像应用领域，用于从 SAR 图像中提取关键点及对应的局部描述子 SAR-SIFT。然而，对于这些局部性特征来说，一个比较关键的问题是如何选取合适的尺度及方向等参数，以对具体的 SAR 图像应用提供充分有效的信息，特别是 SAR 图像分类。

除了这些基本的特征之外，随着机器学习技术的不断发展，大量中层及高层特征也被用于描述 SAR 图像的内容。其中，词袋模型(Bag-of-Words，BoW)作为最简单的中层特征提取方法被广泛应用于描述 SAR 图像。J. Feng 等学者提出一个像素级词袋模型，并通过人工免疫系统对该模型中的各参数进行优化，以实现对 SAR 图像进行表示及分类。通过将 LDA 与 BoW 表示进行结合，R. Bahmanyar 等研究者提出一种主题包模型(Bag-of-Topics，BoT)并用于提取 SAR 图像中的语义级特征。除此之外，针对 SAR 图形的成像特性，稀疏表示也被广泛应用于 SAR 图像及极化 SAR 图像分类。

近年来，随着深度学习(Deep learning，DL)理论被广泛研究及应用，他们也被广泛用于 SAR 图像特征提取。J. Geng 等学者通过一个深度卷积自编码器(Deep Convolutional Auto Encoder，DCAE)对 SAR 图像自动提取特征并用于地物分类操作。该深度卷积自编码器通过一个卷积层提取 SAR 图像中的纹理特征，一个尺度变换层引入邻域信息，四个稀疏自编码层对所提取的纹理特征进行优化及分类及两个后处理层，与其他的手工特征相比而言，该深层结构能从 SAR 图像中自动提取判别特征并用于分类操作。同时，C. Bentes 等研究者提出一个深度神经网络(Deep Neural Network，DNN)对海洋 SAR 图像中的海洋研究设备目标进行检测。通过该深层结构中的隐单元可以直接从原始 SAR 图像数据学习一个层次表示对其进行描述。同时，CNN 也被广泛应用于 SAR 自动目标识别，双极化 SAR 海冰浓度估计等各种应用中的特征提取及学习。J. Ding 学者通过约束特征向量之间的相似性引入监督信息，提出一种基于相似性约束受限玻尔兹曼机(Restricted Boltzmann Machine，RBM)，在对 SAR 图像中的目标进行识别的结果表明该方法能得到比主成分分析(Principal



Component Analysis,PCA)及原始深度置信网络更好的识别性能。

通过将这些从 SAR 图像中提取出的特征,无论是低层的纹理描述子还是中层或者高层特征,结合一定量的标记数据训练监督及半监督分类器,如支撑向量机(Support Vector Machine,SVM),神经网络(Neural Network,NN),随机森林,逻辑回归(Logistic Regression,LR)及 AdaBoost 等对 SAR 图像中不同的地物进行分类。与此同时,在实际应用中,各种 SAR 图像特征也被结合在一起用于 SAR 图像描述,如在多种分类系统中,Gabor 滤波特征和纹理描述子通常被结合在一起以对不同的 SAR 地物获取更为丰富的信息。C. O. Dumitru 等研究者对高分辨 SAR 图像中各种信息提取方法及与 SAR 图像的成像参数之间的关系进行综述及比较。

极化 SAR 理论在全世界得到广泛应用,其应用涉及军用民用等多个方面。我们利用极化 SAR 影像可以获得更多更准确的信息,这些信息可以应用在农林、城市建设、地质灾害预防等多个领域。由此可见,极化 SAR 影像的解译尤为重要。如何实现它是如今遥感领域的热点,而地物分类这一内容则是其中重要的研究内容之一,它将获得的极化 SAR 影像单独提取出来进行研究,快速并且精确地实现极化 SAR 影像地物分类为实现后续解译步骤提供了基础。极化 SAR 影像地物分类首先是获取到极化 SAR 系统从空中观测到的地物信息,然后将每一个像素点进行分类,给予每个像素点不同的类别标记。

12.2 基于深度神经网络的 SAR 影像地物分类

12.2.1 基于自适应自编码和超像素的 SAR 图像分类

本方法首先利用一种新颖的基于 Gamma 分布的异质超像素分割方法对 SAR 图像进行超像素预分割,考虑到了像素间的相关性,将噪声信息的干扰降到最低,然后在预分割的每一小块区域内,基于自适应稀疏自编码器的分类方法,加入图像的多尺度特征,最终利用最近邻聚类算法,进行 SAR 图像分类,并在最后的分类过程中引入了分类门限值和 SAR 图像的均值信息来进一步优化分类结果。

本文所应用的是 Turbopixels 超像素生成方法。它通过自适应局部结构的种子膨胀实现超像素分割,并利用曲率演化模型的骨架化过程,将复杂的超像素分割难题转化为易解的几何流问题,最终将图像分割成网格状的超像素。不但保留了图像的边界信息,还利用紧凑度约束条件限制了图像的过分割,效率也较高。

SAR 图像在边缘分界清晰,匀质或强边界的情况下,超像素能很好地保留图像的边界信息,形成的超像素内包含的像素为同一类地物对象,但在噪声比较严重的弱边界,很难准确地分割,因此算法在弱边界区域形成的超像素内可能包含了两类对象。

SAR 图像的统计模型在地物目标的检测与识别、相干斑噪声的消除、地物目标的分类

等处理中起着非常重要的作用。Gamma 分布符合地球表面大部分地物的统计特征，故 SAR 图像被认为服从 Gamma 分布。Goodman(1975)根据 SAR 图像中相干斑噪声模型，证明了在区域均匀的假设下，SAR 强度图像满足 Gamma 分布。

Gamma 概率分布密度函数为：

$$f(x)=\frac{b^{\upsilon}\cdot x^{\upsilon-1}}{\Gamma(\upsilon)}\cdot \mathrm{e}^{-b\cdot x} \tag{12.24}$$

其中，当 $x\leqslant 0$ 时，$f(x)=0$，$b(b>0)$为 gamma 分布的尺度参数，$\upsilon(\upsilon>0)$为 Gamma 分布的形状参数，也是异质性参数，$\Gamma(\cdot)$是 Gamma 函数；即可以利用 Gamma 分布中的形状参数作为能找出异质超像素的指标。根据极大似然法原理，可推得式(12.24)中 υ、b 的样本估算式。一些具体结果如图 12.9、图 12.10 和表 12.2、表 12.3 所示。

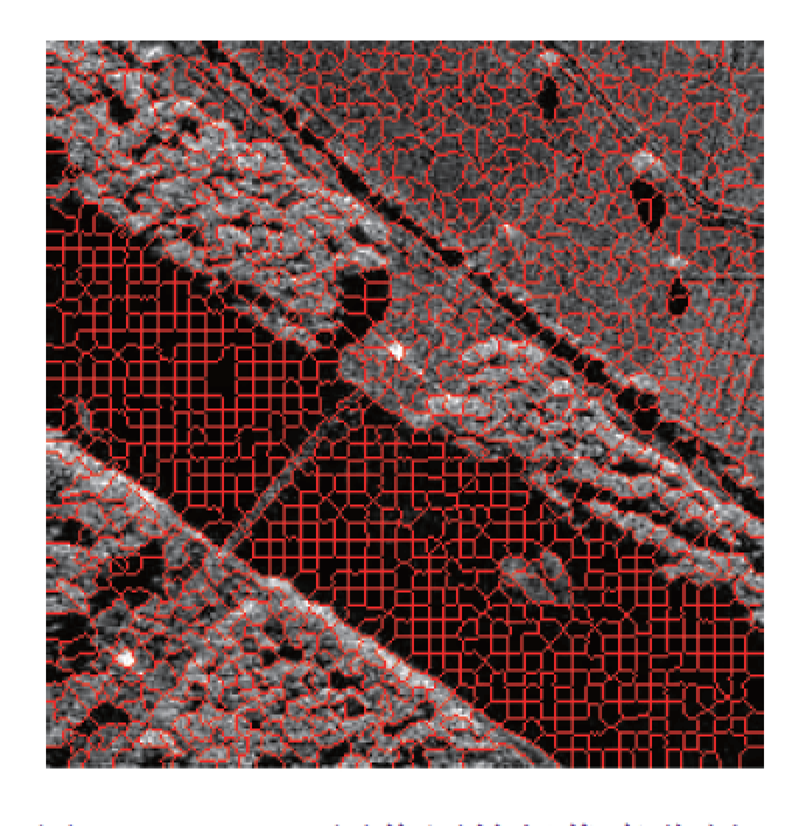


图 12.9　SAR 图像原始超像素分割 1

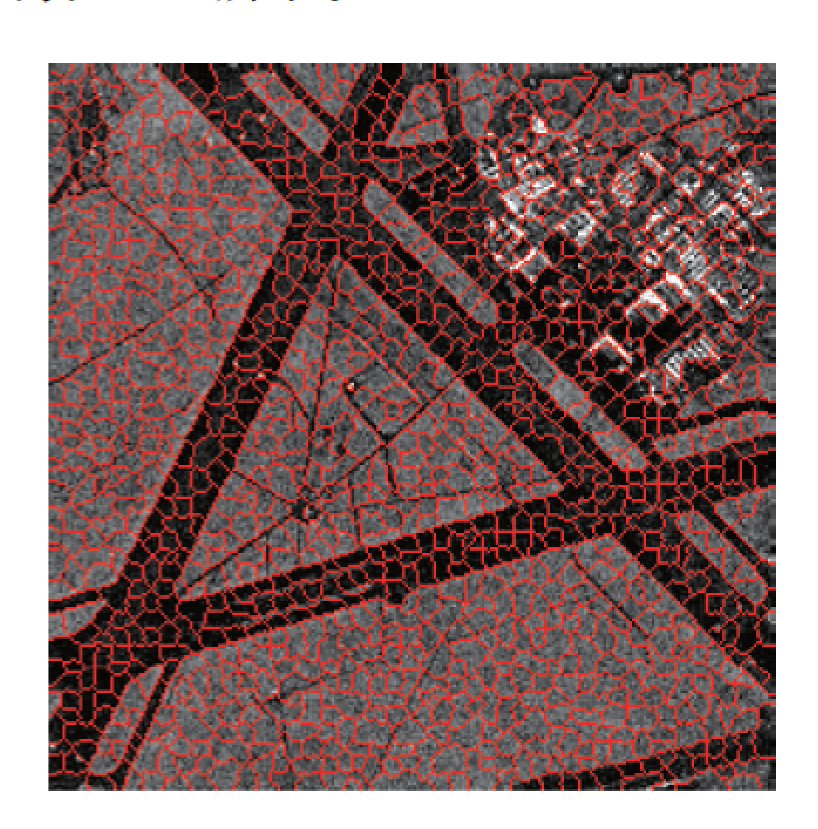


图 12.10　SAR 图像原始超像素分割 2

表 12.2　超像素及其极大似然估计的异质性参数 υ 的展示

编号	(a)	(b)	(c)	(d)	(e)
超像素					
υ	2.2163	1.6238	2.4771	2.6166	1.2751
编号	(f)	(g)	(h)	(i)	(j)
超像素					
υ	6.2724	11.6956	5.5426	7.1817	7.8516

表 12.3　超像素及其极大似然估计的异质性参数 υ 的展示

编　　号	(a)	(b)	(c)	(d)	(e)
超像素					
υ	1.3310	2.3295	2.8023	0.6130	2.5067
编　　号	(f)	(g)	(h)	(i)	(j)
超像素					
υ	1.4450	13.9820	3.7984	14.1388	33.4903

可以看出，表 12.2 和表 12.3 中前五个超像素是异质超像素，异质性系数都比后五个同质超像素小。因此，超像素中像素的异质性系数越小，说明此超像素存在异质即包含两类地物的可能性越大；异质性系数越大，说明存在异质即包含两类地物的可能性越小。

堆栈自编码器可以逐层提取 SAR 图像每个像素深层特征，这些深层特征具有更好的不变性和可分性，为图像分类提供完备的特征需求。但随着 SAR 图像分辨率的提高，相干斑噪声越来越严重，不同地物的异质性相差很大，固定尺度的初始特征取块已经不能满足要求，故提出自适应堆栈自编码的策略。

以一个像素点为中心，$K\times K$ 的窗口尺寸取其邻域，将其拉成$(K\times K)\times 1$ 的特征列向量，以步长为 1 顺序取的整幅 SAR 图像全部像素点的特征向量构造成一组$(K\times K)\times 1$ 的列向量特征作为自编码输入，通过自编码器训练得到一组特征向量。

对每个像素取一固定尺寸的矩形窗 N，对矩形窗内的像素作异质性参数估计，以阈值 m 作为界限，将异质性参数小于 m 的矩形窗（即该区域粗糙度过大）从中心像素扩大到 $N_1(N_1>N)$，异质性参数大于 m 的矩形窗（即该区域粗糙度过小）不改变，即仍为 N，以便所选取的区域能更准确地代表这个区域的特征。

由于自编码器的输入必须是一组维度相等的列向量，所以必须将不同矩形窗拉成的列向量维度统一，故将所有$(N\times N)\times 1$ 的列向量降维到$(N\times N)\times 1$，利用整幅 SAR 图像所有像素点 T 组成一个$(N\times N)\times T$ 维的训练数据集 U。

利用上述基于自适应堆叠自编码的特征提取方法加入多尺度特征信息经 SVM 分类得到的样本集 V 的分类标签结果：Label，对异质超像素过分割的每一个小块 I_j 的标签进行优化，得到每一个小块的最终标签。步骤如下：

(1) 通过自编码特征提取步骤，分别取 L1 和 L2 两个尺度进行自适应堆叠自编码结构

特征提取。

(2) 融合多尺度特征，输入 SVM 分类器对其进行分类得到数据集 V 的分类标签结果。

(3) 根据样本集 V 的分类标签 Label，读取每一个点的标签 Label(i)，Label(i)表示样本集 V 的第 i 个点的标签值。

(4) 对原始 SAR 图像进行异质超像素预分割，可以得到若干超像素：$\{I_1, I_2, \cdots, I_j, \cdots, I_C\}$，$I_j$ 表示 SAR 图像通过异质超像素预分割后的第 j 个超像素，其中 $j=1,2,\cdots,C$，C 表示预分割的超像素数，对于每一个超像素 I_j，统计分布在超像素 I_j 中像素的标签 Label(i)，求得超像素 I_j 中属于每一类标签的像素数。

(5) 在每一个超像素 I_j 中，引入 KNN 算法，k 的大小为每一个超像素内的像素数，根据步骤(4)中得到的每一类标签的像素数，选择超像素 I_j 内像素数最多的那一类的标签 p 作为本超像素的标签。

(6) 将超像素 I_j 内所有像素的标签均重置为 p，则 p 为该超像素的最终标签。

利用三幅 SAR 图像数据来验证本章算法的有效性，对比试验采用加入多尺度特征未加超像素的算法、加入传统超像素算法、多尺度局部模式直方图(MLPH)和局部原始模式(LPP)。实验数据如 12.1.2 节所述，主要从图像的单类精度、总体精度以及 Kappa 系数对比来对本章的算法做评价。本章在所有试验中，基于自适应稀疏自编码器的 SAR 图像分类方法选择两个隐含层的网络，即两层自编码神经网络，我们的第一层和第二层的隐层节点个数相应地分别取为 80 和 64。在输入自适应取块中，我们先取 7×7 的矩阵，异质性的阈值取 0.3，将异质性小于 0.3 的像素选取矩阵扩大到 11×11 经过训练得到一组特征，再取9×9 的矩阵，异质性的阈值取 0.3，将异质性小于 0.3 的像素选取矩阵扩大到 13×13 经过训练得到一组特征。并且根据图像背景图，随机地选择训练数据，将剩余的数据作为我们的测试集，来验证算法的优势。训练数据和测试数据的大小比值大致为 1∶9。

图 12.11(a)是一幅包含了草地、河流和灌木丛的三类复杂地物，特别是灌木丛中有很多阴影，所以纹理比较复杂，为分类处理增加了难度。从对比试验中可以看出，图 12.11(c)是本实验未加超像素用了第 3 章的自适应自编码但加入了多尺度信息所做的分类实验，通过对比发现，加入了本文所提出的异质超像素，使草坪区域的错分孤立点减少了，灌木丛的区域一致性变好了；图 12.11(d)是加入了原始 Turbopixels 超像素的实验，对比可以发现，特别是灌木丛区域，加入了本文所提出的异质超像素，较好地保留了灌木丛的边界轮廓，使其形状完好，且区域连续性增强了；通过与实验图 12.11(e)MLPH 方法对比，该方法较好地保留了原图的细节特点，如河流的中心区域部分。通过与实验图 12.11(f)LPP 方法对比，防止了灌木区对草坪区的影响，降低了草坪区的错分率。表 12.4 给出了算法的准确率对比。

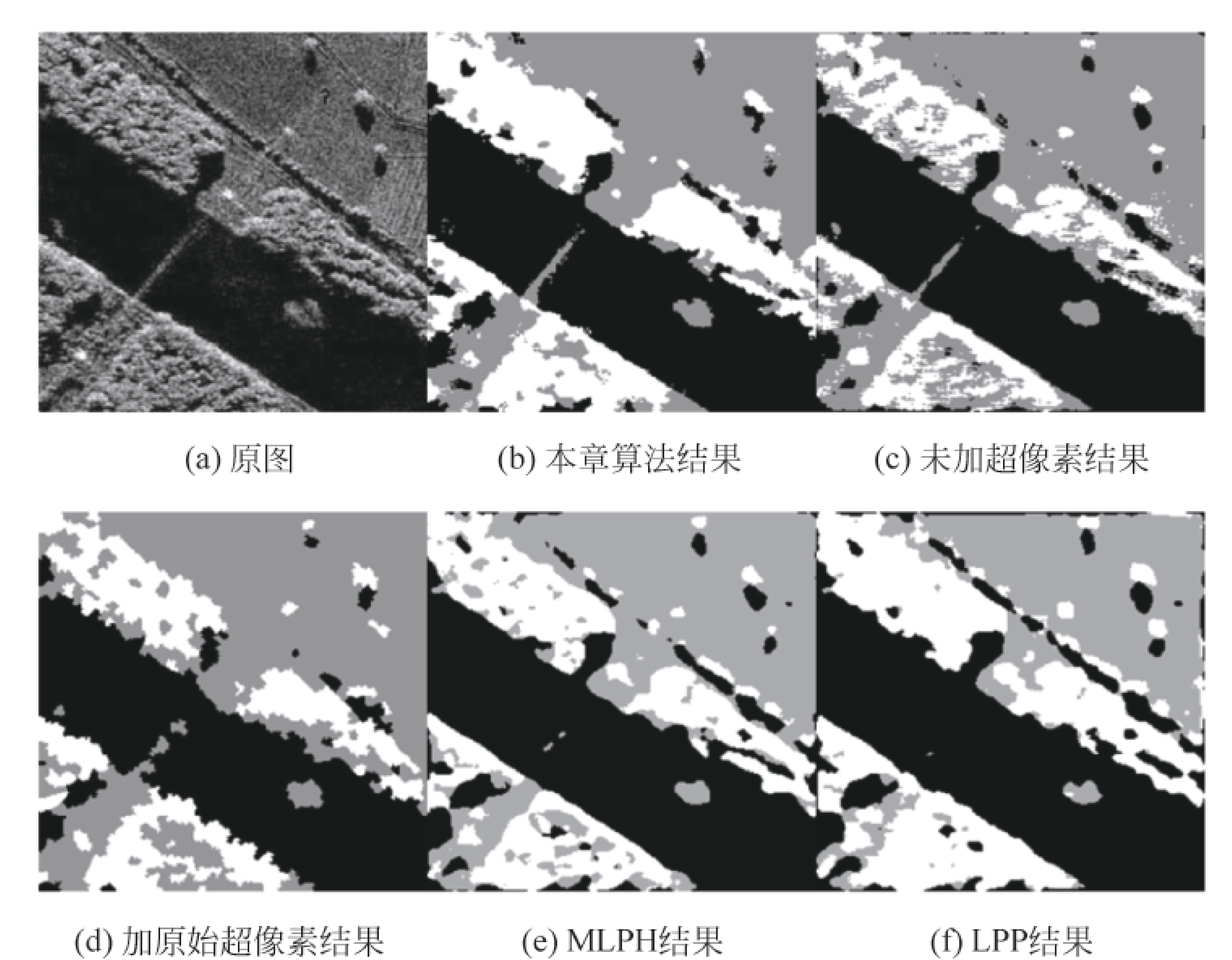
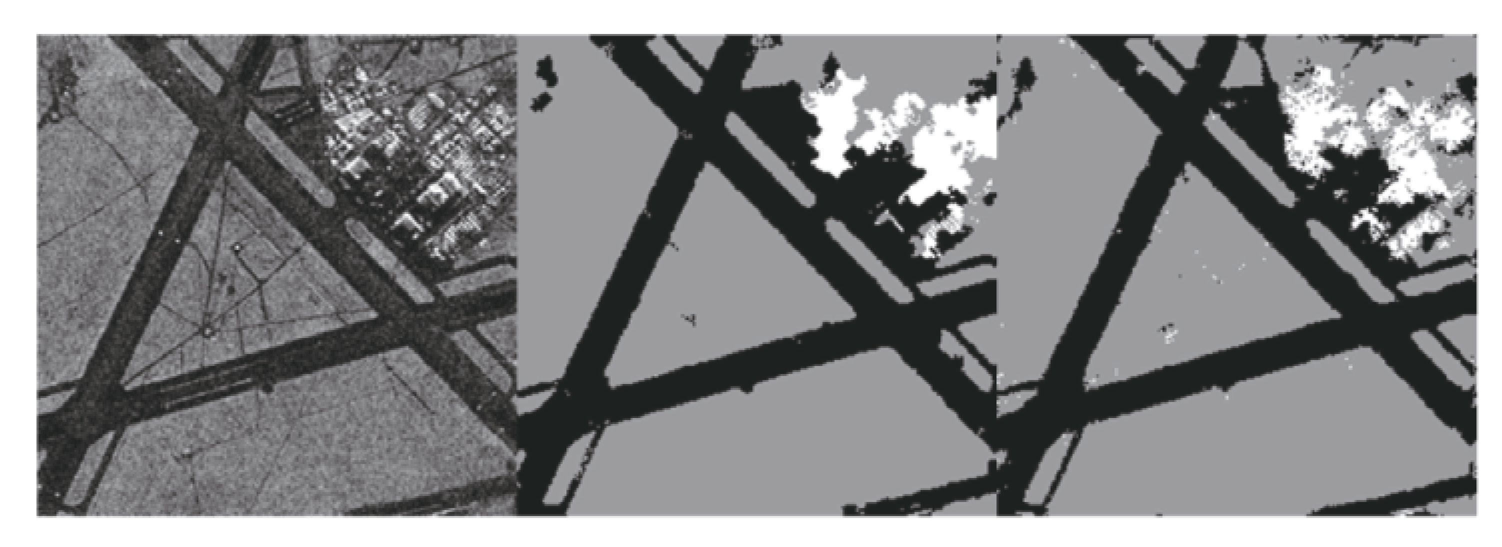


图 12.11　针对 Flevoland 的 SAR 图像分类结果对比图

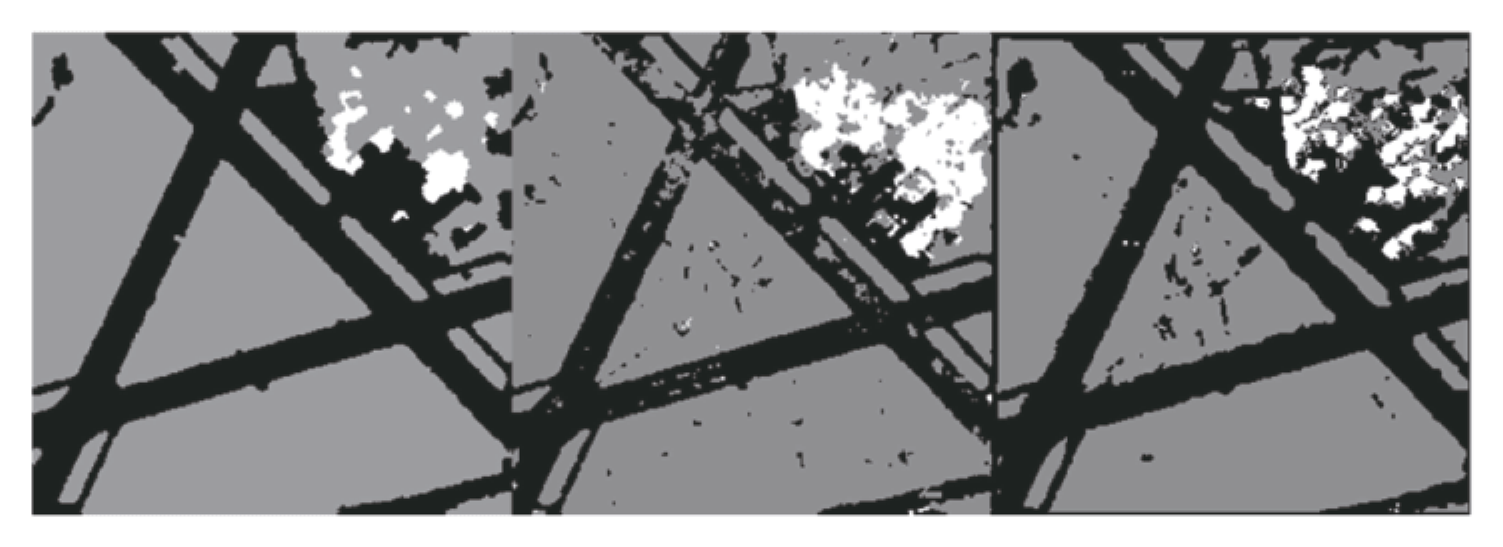
表 12.4　分类算法准确率对比(Flevoland 农田数据)

分类算法	草坪精度	灌木丛精度	河流精度	总体精度	Kappa 系数
本章算法	94.79	89.17	91.18	90.25	0.88
未加超像素	92.48	65.17	89.74	79.39	0.68
加原始超像素	92.73	66.91	85.22	712.14	0.63
MLPH	91.05	77.32	84.32	81.03	0.69
LPP	78.66	812.14	89.73	82.44	0.70

图 12.12(a)是一幅包含了草地、轨道和城市区域的三类复杂地物,特别是城市区域的纹理比较复杂,是 SAR 图像中较为难分的一类地物。从对比试验中可以看出,图 12.12(c)是本实验未加超像素用了第 3 章的自适应自编码但加入了多尺度信息所做的分类实验,通过对比发现,加入了本文所提出的异质超像素,使草坪区域的错分孤立点减少了,有效地抑制了噪声的影响,使城市区域一致性变好了,比较完好地保留了城市区的形状及细节特征;图 12.12(d)是加入了原始 Turbopixels 超像素的实验,对比可以发现,特别是城市区域,加入了本文所提出的异质超像素,较好地保留了城市的边界轮廓,使其形状完好,且区域连续性增强了;通过与实验图 12.12(e)MLPH 方法对比,本章方法较好地抑制了噪声的影响,增强了均匀区域的连续性。通过与实验图 12.12(f)LPP 方法对比,本章算法对城市区域形状保留完好,且降低了草坪区的错分率。表 12.5 给出了各类算法的准确率对比。



(a) 原图　(b) 本章算法结果　(c) 未加超像素结果



(d) 加原始超像素结果　(e) MLPH结果　(f) LPP结果

图 12.12　对 Germany 数据的 SAR 图像分类结果对比图

表 12.5　分类算法准确率对比(Germany 数据)

分类算法	城市精度	跑道精度	草坪精度	总体精度/%	Kappa 系数
本章算法	62.37	90.47	96.49	93.62	0.90
未加超像素	58.64	90.10	94.72	89.40	0.82
加原始超像素	24.30	90.77	95.43	88.01	0.81
MLPH	65.95	81.08	95.75	83.88	0.73
LPP	43.30	85.17	82.46	80.35	0.69

图 12.13(a)是一幅包含了草地、轨道和灌木丛的三类复杂地物，特别是灌木丛中有很多阴影，所以纹理比较复杂，为分类处理增加了难度。从对比试验中可以看出，图 12.13(c)是本实验未加超像素用了第 3 章的自适应自编码并加入了多尺度信息所做的分类实验，通过对比发现，加入了本节所提出的异质超像素，使草坪区域的错分孤立点减少了，灌木丛的区域一致性变好了且对轨道的形状保留完好；图 12.13(d)是加入了原始 Turbopixels 超像素的实验，对比可以发现，特别是灌木丛区域，加入了本节所提出的异质超像素，较好地保留了灌木丛的边界轮廓，使其形状完好，且区域连续性增强了；通过与实验图 12.13(e)MLPH 方法对比，本章方法有效地抑制了灌木丛区对草坪区的影响，减少了草坪区和轨道区的错分孤立点。通过与实现图 12.13(f)LPP 方法对比，本章算法增强了灌木丛区的区域一致性。表 12.6 列出了各类算法的准确率对比。

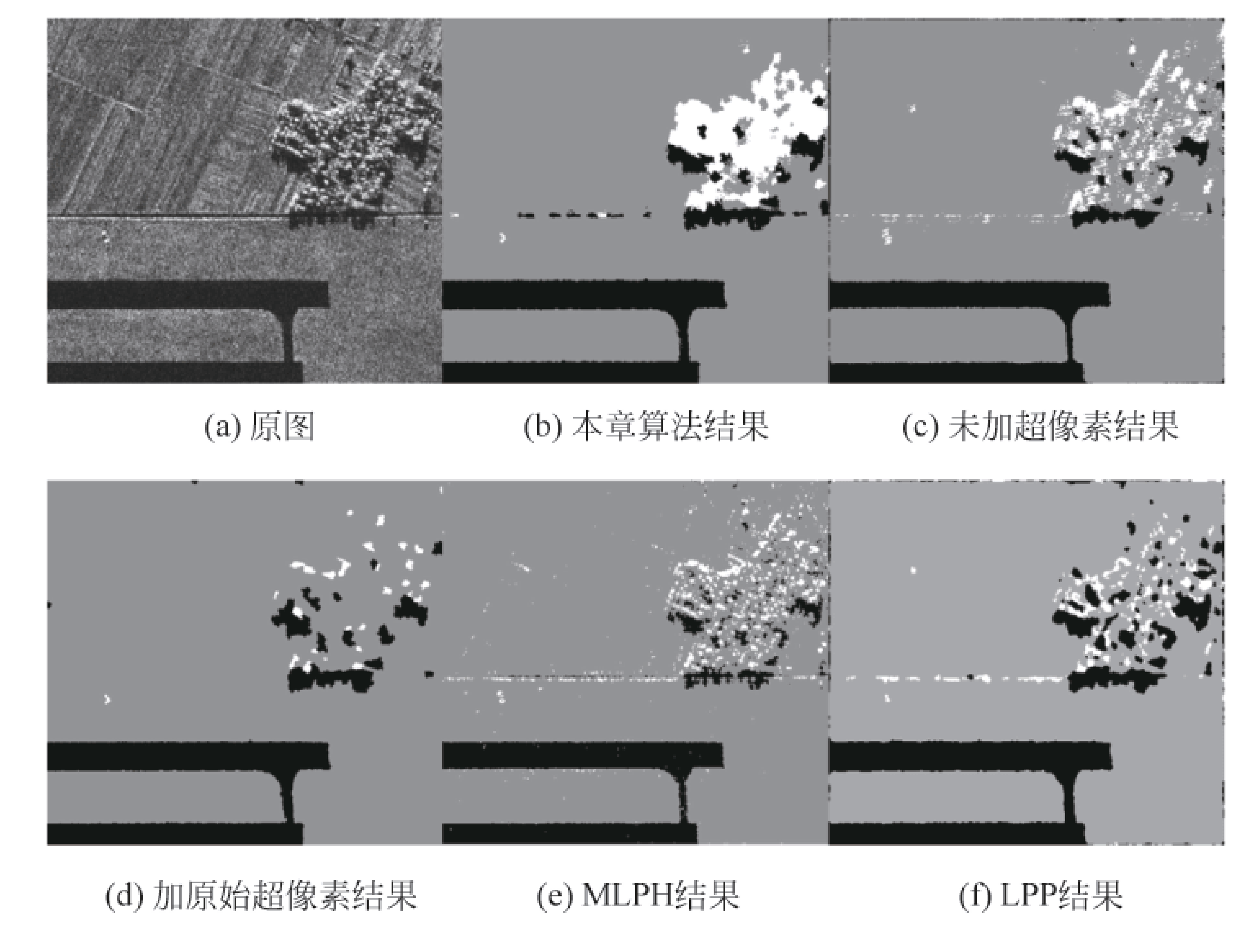


图 12.13　对 San Francisco 数据的 SAR 图像分类结果对比图

表 12.6　分类算法准确率对比(San Francisco 数据)

分类算法	草坪精度	灌木丛精度	跑道精度	总体精度	Kappa 系数
本章算法	912.27	62.54	89.54	93.61	0.91
未加超像素	97.88	29.67	81.35	88.05	0.82
加原始超像素	93.72	7.97	87.49	80.38	0.82
MLPH	92.54	16.76	84.44	77.43	0.62
LPP	97.13	18.03	87.83	812.45	0.83

本方法的区域内错分孤立的点很少，一致性比较好，并且边缘清晰，细节信息保存完整，根据本章的算法得到的分类效果图能准确地对 SAR 图像的地物进行分类，边缘轮廓清晰准确，从而证明了基于自适应堆叠自编码和异质超像素的 SAR 图像分类算法的有效性。应用本方法与采用加入多尺度特征未加入超像素的算法、加入传统超像素算法、MLPH 和 LPP。进行实验分类精度对比，从多个方面观测自适应堆叠自编码器可以自动提取有效的特征，本章提出的方法的实验结果误分率明显变小，从视觉效果上看出本算法消除了边缘不整齐的问题，每一块区域的内部没有错分的点，比上一章基于自适应堆叠自编码的初始分割结果效果更加理想，这一点可以从图 12.12 的城市区域、图 12.11 和图 12.13 的灌木丛区域明显看出。

12.2.2　基于卷积中层特征学习的 SAR 图像分类

SAR 图像分类技术是对单幅图像中的不同地物进行分类，词袋模型应用于 SAR 图像

分类问题中必须首先将单幅 SAR 图像分割成几百或上千个极小的匀质区域，因此初始预分割技术的边缘保持性将严重影响最终的分类结果，同时由于每个极小的匀质区域像素数量有限，提取的直方图特征很难具有高度的判别性。受卷积神经网络的启发，提出基于卷积的中层特征提取方法。将每个像素的特征与视觉单词进行卷积后最大值池化作为中层特征，改进了经典词袋模型应用于 SAR 图像时的不足，获得了良好的分类效果。

研究已经表明人类视觉系统对具体方向和空间频率都比较敏感。图像的频率信息能够反映图像灰度信息的变化程度。实验表明应用 Gabor 滤波器实部进行滤波能够实现图像平滑，应用 Gabor 滤波器虚部进行滤波能够实现边缘检测。

通过设置 Gabor 函数的参数值，我们可以获得不同形状的滤波器，进而抓取不同的纹理结构。对二维 Gabor 小波函数进行傅里叶变换得

$$G(u,v) = \exp\left\{-\frac{1}{2}\left[\frac{(u-W)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right]\right\} \tag{12.25}$$

其中 $\sigma_u=\dfrac{1}{2\pi\sigma_x}$，$\sigma_v=\dfrac{1}{2\pi\sigma_y}$以 $g(x,y)$为母函数，对其进行不同尺度的旋转、变换和膨胀，可以得到一系列完备的非正交基，因此 Gobor 小波变换为

$$g_{mn}(x,y) = a^{-m}g(x',y'),\quad a>1,m,n\in\mathbb{N} \tag{12.26}$$

$$x' = a^{-m}(x\cos\theta+y\sin\theta),\quad y' = a^{-m}(-x\sin\theta+y\cos\theta) \tag{12.27}$$

$$u' = a^{-m}(u\cos\theta+v\sin\theta),\quad v' = a^{-m}(-u\sin\theta+v\cos\theta) \tag{12.28}$$

其中 $\theta=n\pi/k$，参数 k 为选择的不同方向的数目，参数 m 为相应的不同尺度，参数 n 表示不同的方向。

对一幅图像 $I(x,y)$进行 Gabor 小波变换可得

$$W_{mn}(x,y) = \iint I(x,y)\cdot g_{mn}^*(x-x_1,y-y_1)\mathrm{d}x_1\mathrm{d}y_1 \tag{12.29}$$

其中 $g_{mn}^*(x-x_1,y-y_1)$表示基函数 $g_{mn}(x-x_1,y-y_1)$的共轭复数。经过不同尺度和方向的变换后，图像的均值 μ_{mn} 和方差 σ_{mn} 分别为

$$\mu_{mn} = \iint W_{mn}(x,y)\mathrm{d}x\mathrm{d}y \tag{12.30}$$

$$\sigma_{mn} = \sqrt{\iint (W_{mn}(x,y)-\mu_{mn})^2\mathrm{d}x\mathrm{d}y} \tag{12.31}$$

最终由图像的均值 μ_{mn} 和方差 σ_{mn} 构成的纹理特征向量为

$$f = (\mu_{00},\sigma_{00},\mu_{01},\sigma_{01},\cdots,\mu_{m-1n-1},\sigma_{m-1n-1}) \tag{12.32}$$

传统的字典学习方法是应用一种无监督的学习方式，如 K-均值对从图像中采样得到的训练样本进行聚类。K-均值仅适用于具有少数几类简单匀质区域的图像，但是对于具有复杂纹理信息的 SAR 图像，并不能保证获得一个最优的视觉词典。K-均值算法训练得到的词典不具有确定性和合理性。为改进 K-均值算法而提出了 K-SVD 算法。K-SVD 算法为对 K-均值算法的推广，当 K-SVD 算法中仅由一个聚类中心表示样本点时即为 K-均值算法。



K-SVD 算法是一个学习字典的优化算法。其优化学习的目标函数为

$$\min_{\boldsymbol{D},\boldsymbol{X}} \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2$$
$$\text{s.t.} \|\boldsymbol{X}(:,i)\|_0 \leqslant T_0, i=1,2,\cdots,N \tag{12.33}$$

其中 $\boldsymbol{Y}$ 是为算法输入的无标记样本，$\boldsymbol{D}$ 为待训练的过完备稀疏冗余字典，$\boldsymbol{X}$ 为应用字典 $\boldsymbol{D}$ 对样本 $\boldsymbol{Y}$ 进行表示的稀疏系数，通过 K-SVD 算法优化得到该目标函数的最小值。$\boldsymbol{X}$ 中大部分值接近或等于 0，因此为稀疏表示过程。

通过已知的训练样本 $\boldsymbol{Y}$ 优化得到完备冗余字典 $\boldsymbol{D}$ 和稀疏表示系数 $\boldsymbol{X}$ 将是一个非凸的病态问题。因此一般的优化学习方法将是一个求解局部最优值的过程。K-SVD 算法首先固定字典 $\boldsymbol{D}$，优化稀疏表示 $\boldsymbol{X}$，因此问题转化为一个普通进行稀疏表示的问题。初始时字典 $\boldsymbol{D}$ 为随机初始化或应用指定的训练样本进行初始化。常用的稀疏表示算法为正交匹配追踪算法(OMP)，通过字典 $\boldsymbol{D}$ 优化得到稀疏表示系数 $\boldsymbol{X}$。然后由得到的稀疏表示系数 $\boldsymbol{X}$ 来更新优化字典 $\boldsymbol{D}$，式(12.34)为字典 $\boldsymbol{D}$ 的优化过程。

$$\begin{aligned}\|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2 &= \left\|\boldsymbol{Y}-\sum_{j=1}^{K}\boldsymbol{d}_j\boldsymbol{x}_T^j\right\|_F^2 \\ &= \left\|\left(\boldsymbol{Y}-\sum_{j\neq k}\boldsymbol{d}_j\boldsymbol{x}_T^j\right)-\boldsymbol{d}_k\boldsymbol{x}_T^k\right\|_F^2 \\ &= \|\boldsymbol{E}_k-\boldsymbol{d}_k\boldsymbol{x}_T^k\|_F^2\end{aligned} \tag{12.34}$$

其中 K 为字典 $\boldsymbol{D}$ 的总的字典原子的个数，即字典 $\boldsymbol{D}$ 共有 K 列，每一列为一个字典原子；d_k 为字典 $\boldsymbol{D}$ 的第 k 列，即字典 $\boldsymbol{D}$ 中第 k 个字典原子；$\boldsymbol{E}_k$ 为字典原子 $\boldsymbol{d}_k$ 的残差。K-SVD 通过 SVD 奇异值分解方法优化每个字典原子从而减小误差获得最优值。

词袋模型中特征编码部分为构建视觉词典，并以视觉单词为基底提取直方图特征。特征编码步骤是一个非常复杂，并且扁平的单层操作。受卷积特征提取的启发，将低层提取的特征与视觉词典中每个视觉单词进行卷积，获得卷积特征作为词袋模型中的特征编码步骤。视觉词典的构建应用 K-SVD 算法进行有监督训练，由于 SAR 图像地物分布的不确定性，导致不同地物之间的像素数量差异悬殊。例如一幅 SAR 图像中大部分面积是农田，只有小部分为城市，则 K-SVD 进行初始化时，初始化样本将大部分为农田样本，很小一部分样本为城市，样本的分布不均衡将直接导致训练的视觉词典缺乏代表性和普适性。为防止 K-SVD 算法初始化时样本分布不均衡而陷入局部最优，将原来的随机初始化修改为有监督地对每类地物抽取相同数量级的像素数进行初始化。由于 K-SVD 训练的视觉词典中不同视觉单词之间能够对初始样本进行稀疏冗余表示，因此每个视觉单词均有一定的代表性。如图 12.14 所示为基于卷积特征学习的模型的框架结构，低层特征提取步骤为：对待分类的 SAR 图像进行不同尺度、不同方向上的 Gabor 小波变换，并以 9×9 的邻域内的均值和方差作为每个像素点的低层特征向量；应用 K-SVD 算法建立视觉词典，同时将每个像素点的低层 Gabor 特征向量与视觉词典做卷积，并通过最大值池化来降低每个像素点的中层卷积特征的维度，得到中层卷积特征；最后将中层卷积特征输入到 SVM 分类器中对每个像素点进

行分类，获得其类别标 SVM 分类器中对每个像素点进行分类，获得其类别标签。

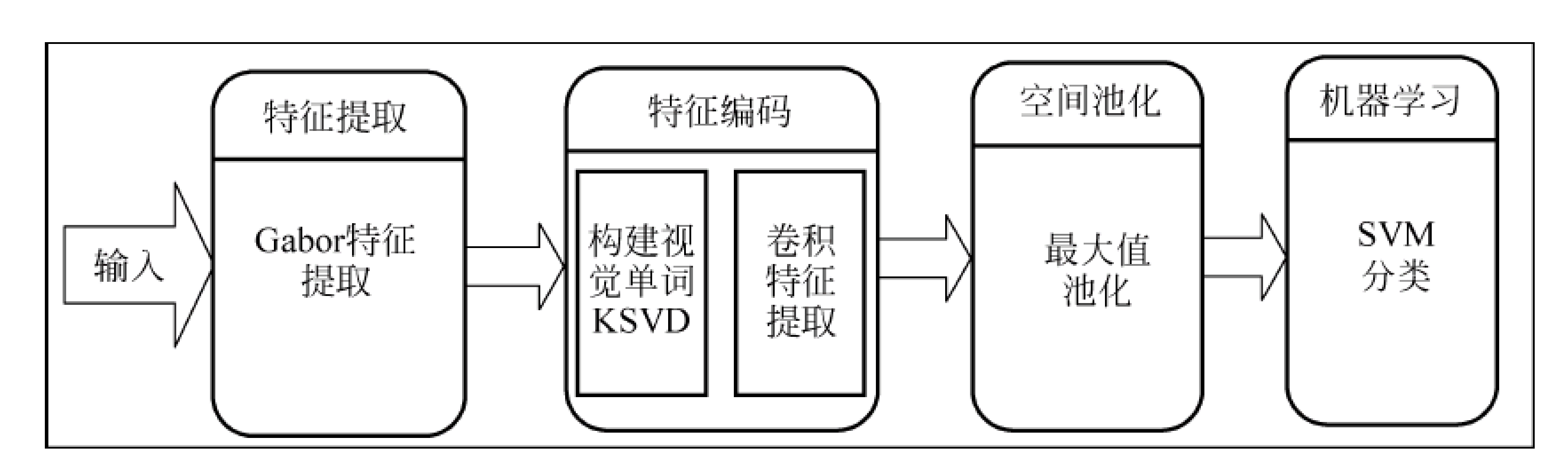


图 12.14　卷积词袋模型框架结构图

首先对输入的 SAR 图像分别在六个尺度（$f=7.8769$，$f=4.5310$，$f=4.0960$，$f=3.9084$，$f=3.5804$，$f=2.6806$），三个方向（$\theta=0°$，$\theta=60°$，$\theta=120°$）进行加窗傅里叶（Gabor）变换，每个像素点取其 9×9 邻域中在不同尺度和方向的变换域内的均值和方差作为其低层 Gabor 特征 $\boldsymbol{F}_1=[\boldsymbol{I}_1;\boldsymbol{I}_2;\cdots;\boldsymbol{I}_N]$，其中 N 表示输入的 SAR 图像中的像素个数，$\boldsymbol{I}_n$ 表示每个像素点提取的特征向量，$n\in[1,2,\cdots,N]$。显然 $\boldsymbol{I}_n$ 的维度大小为 1×36。通过 K-SVD 算法训练得到视觉词典。首先对字典 $\boldsymbol{D}$ 进行有监督的初始化，不同类别的地物采样同数量级的像素的低层 Gabor 特征来初始化字典 $\boldsymbol{D}$，有效避免了初始样本分布不均衡的可能。然后应用 K-SVD 算法训练一个过完备的稀疏冗余字典 $\boldsymbol{D}$。该过程的具体步骤如下：

（1）对每类地物的低层特征 Gabor 特征进行随机采样，得到训练样本，每类地物的训练样本的特征向量维度是 50×36；

（2）选择前 K 个像素点的特征向量来初始化字典 $\boldsymbol{D}$；

（3）固定字典 $\boldsymbol{D}$，利用正交匹配追踪算法对字典 $\boldsymbol{D}$ 进行稀疏表示，得到稀疏表示稀疏 $\boldsymbol{X}$；

（4）根据系数矩阵 $\boldsymbol{X}$ 对字典 $\boldsymbol{D}$ 进行 K 次迭代，更新字典 $\boldsymbol{D}$，更新规则为优化目标函数：

$$\begin{aligned}\min_{\boldsymbol{D},\boldsymbol{X}}\|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2&=\min_{\boldsymbol{D},\boldsymbol{X}}\left\|\boldsymbol{Y}-\sum_{j=1}^{K}\boldsymbol{D}_j\boldsymbol{x}_T^j\right\|_F^2\\&=\min_{\boldsymbol{D},\boldsymbol{X}}\left\|\left(\boldsymbol{Y}-\sum_{j\neq k}\boldsymbol{D}_j\boldsymbol{x}_T^j\right)-\boldsymbol{D}_k\boldsymbol{x}_T^k\right\|_F^2\\&=\min_{\boldsymbol{D},\boldsymbol{X}}\|\boldsymbol{E}_k-\boldsymbol{D}_k\boldsymbol{x}_T^k\|_F^2\end{aligned}\tag{12.35}$$

每次迭代对进行奇异值（SVD）分解并更新 $\boldsymbol{D}_k$，使得目标函数值最小，其中 E_k 去掉字典 $\boldsymbol{D}_k$ 的重构误差，T 为应用 OMP 进行稀疏表示得到稀疏系数 $\boldsymbol{X}$ 中非零元素的个数的最大值。通过阈值 T 的设置来达到稀疏表示的目的。

将视觉词典 $\boldsymbol{D}$ 与低层 Gabor 特征进行卷积与池化，得到中层卷积特征。由于每个像素点的低层 Gabor 特征向量分别与字典 $\boldsymbol{D}$ 中每个字典原子进行卷积，每个像素点的特征从 1×36 维变化为 K×36 维，由于每个像素点的特征维度变大，并且考虑图像的静态特性，可

以对由卷积得到的特征进行最大值聚合，降低特征维度，最终得到中层卷积特征 F_2。

接着应用 SVM 预测场景类标。SVM 分类器中核函数设置为高斯核函数，首先对深层特征 F_2 进行随机采样训练 SVM 模型参数；然后将全部像素点的深层特征 F_2 输入已训练好的 SVM 分类模型中进行分类，获得分类结果 R_1，共分为 C 类场景。

图 12.15(a)为一幅大小为 256×256 的 Radarsat 水域 SAR 图像。该 SAR 图像共包含两类简单地物：河流、陆地。图 12.15(b)为 K-BOW 方法对图 12.15(a)进行分类得到的结果。由图 12.15(b)可得 K-BOW 方法应用于该 SAR 图像进行分类时，受噪声影响严重，将许多陆地区域错分为了河流，分类准确率较低。图 12.15(c)为 GMM-BOW 方法对图 12.15(a)进行分类得到的结果。由图 12.15(c)可得 GMM-BOW 方法较 K-BOW 有了一定的改进，但背景和河流的边界分类模糊，这一结果是由于词袋模型中层直方图特征未能有效区分不同类别地物的原因。图 12.15(d)为本章所提出的算法对图 12.15(a)进行分类得到的结果。由图 12.15(d)可得本章节提出的方法准确率最高，且区域一致性效果最好，边缘保持良好。由表 12.7 可知本章提出的改进算法的分类结果从准确率和 Kappa 系数两个评价指标方面均有所提升。

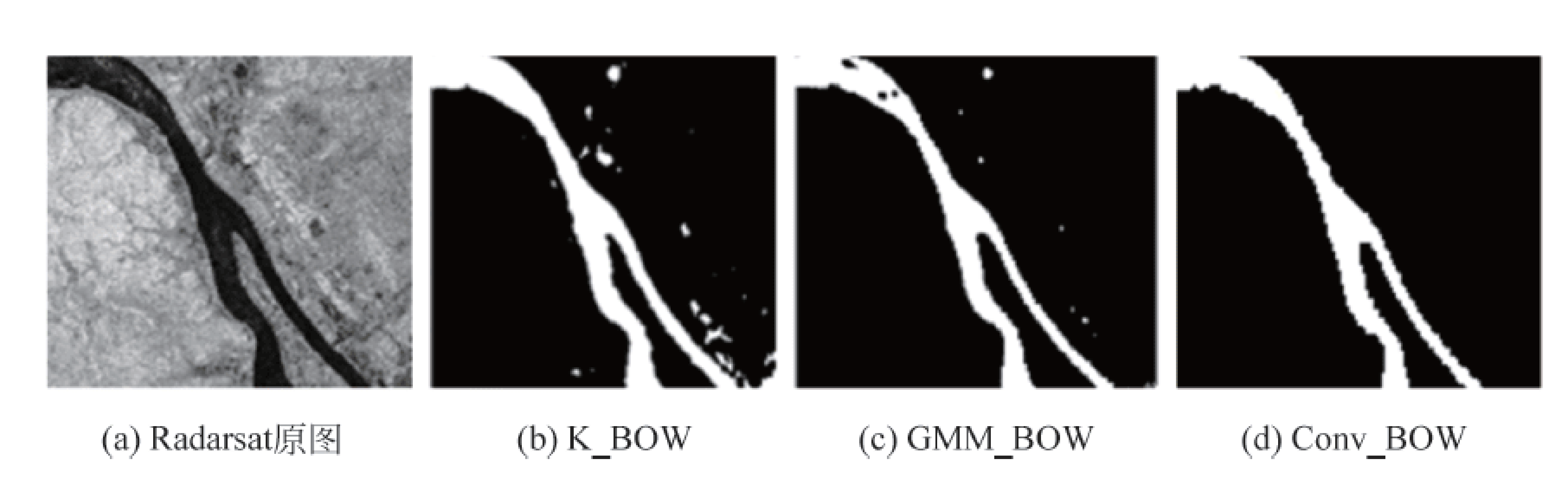


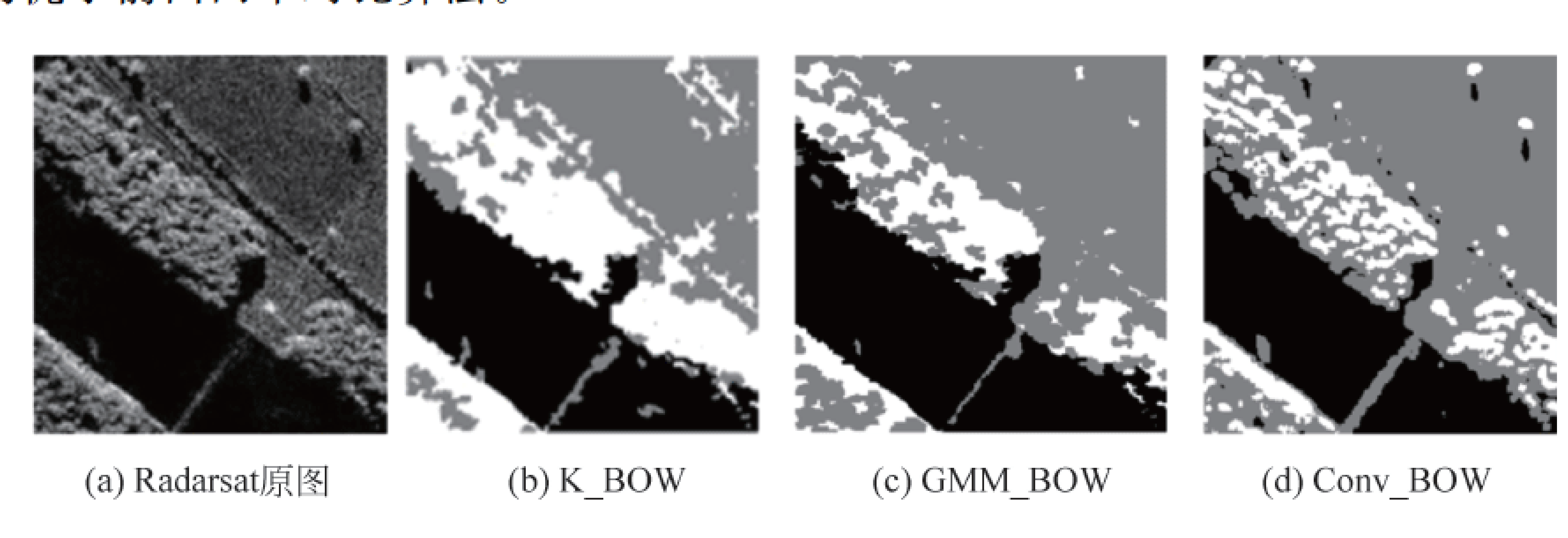
图 12.15　Radarsat 水域 SAR 图像分类结果

表 12.7　Radarsat 水域 SAR 图像分类结果评价指标

算　　法	K-BOV	GMM-BOV	Conv-BOV
准　确　率	94.81	96.58	97.17
Kappa 系数	0.9059	0.9364	0.9541

图 12.16(a)为一幅位于美国新墨西哥州 Albuquerque 地区附近的 RioGrande river 区域，分辨率为 1m，大小为 256×256 的 Ku 波段 SAR 图像。该 SAR 图像中共包含三类地物：农田、植被及河流。图 12.16(b)为 K-BOW 方法对图 12.16(a)进行分类得到的结果。由图 12.16(b)可得 K-BOW 方法应用于该 SAR 图像进行分类时，受噪声影响严重，植被与农田区域边界分类模糊，边缘不够平滑，且将许多农田区域错分为了植被，存在严重的错分现象。图 12.16(c)为 GMM-BOV 方法对图 12.16(a)进行分类得到的结果。由图 12.16(c)可得 GMM-BOV 算法应用于该 SAR 图像进行分类时，将大量植被错分为了农田区域，即 GMM-BOW 方法比较适合于分类纹理简单的地物，如农田、河流等，但对于含有复杂纹理区

域的图像，分类结果较差，因此该算法应用于分类时，存在一定的局限性。图 12.16(d)为本章节算法对图 12.16(a)进行分类得到的结果。由图 12.16(d)可得本章提出的算法对含有植被复杂纹理的 SAR 图像分类精确，能够正确区分植被及其周围的阴影区域，对小目标提取完整。同时与 K-BOW 算法和 GMM-BOW 算法相比，本章节提出的算法能够更好地保持区域一致性，不同地物交界处边缘分类精确，无毛刺现象。可见基于卷积特征学习的 SAR 图像方法不仅能够有效区分农田、河流等简单地物，对于植被等复杂地物同样具有优良的分类性能。由表 12.8 可知本章提出的改进算法的分类结果在准确率和 Kappa 系数两方面均优于前面两个对比算法。



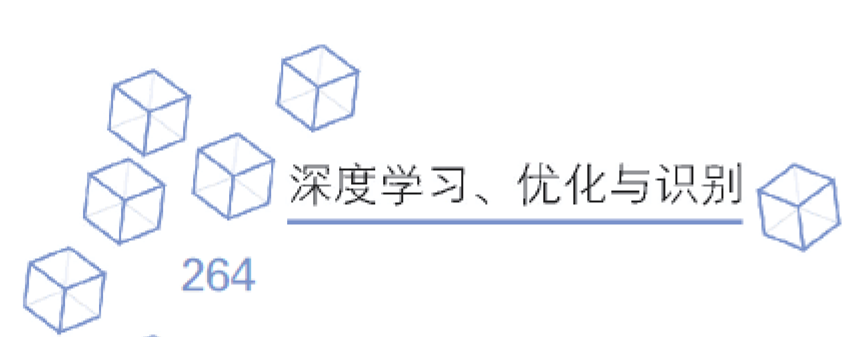
(a) Radarsat原图　(b) K_BOW　(c) GMM_BOW　(d) Conv_BOW

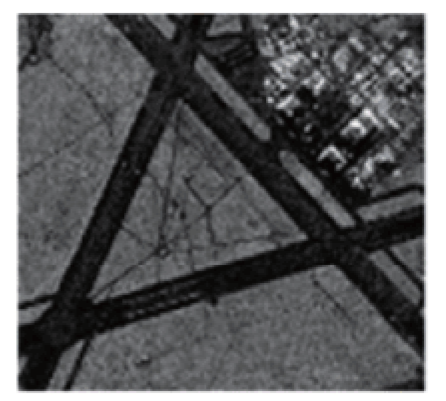
图 12.16　RioGrande river 水域 SAR 图像分类结果

表 12.8　RioGrande river 水域 SAR 图像分类结果评价指标

算　　法	K-BOV	GMM-BOV	Conv-BOV
准　确　率	91.53	93.62	94.37
Kappa 系数	0.8194	0.8372	0.8526

图 12.17(a)为一幅位于美国加州某一区域，分辨率为 3m，大小为 256×256，Ku 波段 SAR 图像。该图共包含三类地物：跑道、路面和建筑。图 12.17(b)为 K-BOW 方法对图 12.17(a)进行分类得到的结果。由图 12.17(b)可得 K-BOW 方法应用于该 SAR 图像进行分类时，跑道边缘分类模糊，边缘分类不够精确，在跑道与路边的交界处，将路面错分为了跑道，且忽略了一些细小纹理，同时将建筑周围的阴影错分为了建筑，存在严重的错分现象，从而影响了分类准确率。图 12.17(c)为 GMM-BOW 方法对图 12.17(a)进行分类得到的结果。由图 12.17(c)可得 GMM-BOW 方法应用于该 SAR 图像进行分类时，跑道区域受噪声影响严重，存在严重的错分现象，且在跑道与路面的交界处附近，将跑道错分为了路面，同时对图像中的细小纹理分类错误，分类效果较差。图 12.17(d)为本章提出的方法对图 12.17(a)进行分类得到的结果。由图 12.17(d)可得本章提出的方法应用于该 SAR 图像进行分类时，弱边缘分类正确，小目标提取完整，对含有跑道复杂纹理的 SAR 图像能够保持较好的区域一致性。由表 12.9 可知本章提出的改进算法的分类结果在准确率和 Kappa 系数两方面均取得了最好的统计值。




(a) 加州原图


(b) K_BOW


(c) GMM_BOW


(d) Conv_BOW

图 12.17　美国加州某区域 SAR 图像分类结果

表 12.9　美国加州某区域 SAR 图像分类结果评价指标

算　　法	K-BOV	GMM-BOV	Conv-BOV
准　确　率	93.89	94.76	95.83
Kappa 系数	0.8769	0.8873	0.9217

综上所述，本章提出的基于卷积特征学习的 SAR 图像分类方法中提取的卷积特征能够弱化噪声的同时增强原有的数据结构，同时能够抓取图像的细节纹理，边缘分类清晰同时保持良好的匀质性，有效提高了分类准确率。

12.3　基于第一代深度神经网络的 PolSAR 影像地物分类

12.3.1　基于稀疏极化 DBN 的极化 SAR 地物分类

1. 模型介绍——稀疏极化 DBN 模型的构建

构造基于稀疏 DBN 的四层稀疏深度网络，包括一个输入层，两个隐含层和一个分类层，最后的分类层使用人工神经网络（NN）模型产生最终的网络输出。可视层的节点数是由数据的基础特征数决定的，基础特征可根据上述的特征提取阶段得到，即为 300，也就是说稀疏 DBN 的可视层节点数为 300；第二层和第三层隐含层节点数分别为 150、100，这是经过多次实验得到的比较好的隐层节点数；输出层是针对极化 SAR 地物分类的 NN 模型，因此输出节点数为数据的类别数。稀疏极化 DBN 学习本章中使用对比散度算法对稀疏深度网络的前三层进行逐层预训练，计算两个隐含层的输出值和各层之间的权值及偏置。下面将给出本章算法对极化 SAR 地物分类的详细分类步骤：

（1）输入极化 SAR 数据的特征，根据 12.2 节计算特征；

（2）构建稀疏极化 DBN，并确定各个参数 $\theta=\{\boldsymbol{W},\boldsymbol{a},\boldsymbol{b}\}$，$\boldsymbol{W},\boldsymbol{a},\boldsymbol{b}$ 分别是权值、可视层偏置和隐含层偏置；

（3）对可视层输入 $\boldsymbol{v}_i^0$ 进行正向传播，计算出隐含层的输出 $\boldsymbol{h}_j^0$；

（4）对隐含层的输出 $\boldsymbol{h}_j^0$ 进行反向传播，得到 $\boldsymbol{v}_i^1$；

(5) 结合参数,更新模式参数 $\theta=\{\boldsymbol{W},\boldsymbol{a},\boldsymbol{b}\}$,参数的变化量是:

$$\Delta W_{ij}=\lambda_w\cdot(\mathbb{E}[\boldsymbol{v}_j^0\cdot\boldsymbol{h}_j^0]-\mathbb{E}[\boldsymbol{v}_j^1\cdot\boldsymbol{h}_j^1])+\lambda\cdot\sum_{l=1}^{m}\boldsymbol{h}_j^l\cdot(1-\boldsymbol{h}_j^l)\cdot\boldsymbol{v}_i^l \tag{12.36}$$

$$\Delta a_i=\lambda_a\cdot(\mathbb{E}[\boldsymbol{v}_j^0]-\mathbb{E}[\boldsymbol{v}_j^1])+\lambda\cdot\sum_{l=1}^{m}\boldsymbol{h}_j^l\cdot(1-\boldsymbol{h}_j^l) \tag{12.37}$$

$$\Delta b_j=\lambda_b\cdot(\mathbb{E}[\boldsymbol{h}_j^0]-\mathbb{E}[\boldsymbol{h}_j^1]) \tag{12.38}$$

(6) 重复计算步骤(3)～(5),直至收敛;

(7) 将上述得到的 $\boldsymbol{v}_i^1$ 作为下一次的可视层 $\boldsymbol{v}_i^0$,然后重复步骤(3)～(7),直至各层训练结束;

(8) 根据 NN 模型,对稀疏极化 DBN 进行预训练;

(9) 使用反向传播算法的有限内存的 BFGS(LBFGS)算法对整个稀疏深度网络进行微调,优化分类网络中的各种参数,完成稀疏深度网络的训练;

(10) 利用训练好的稀疏极化 DBN 对待分类的极化 SAR 数据地物分类,得到极化 SAR 地物分类结果。

2. 实验结果与分析

1) Flevoland 农田数据实验的结果

实验随机选取了每类 1000 个样本作为训练样本,剩余的样本均为测试样本。此幅图像分辨率较低,类别数较多,每一类的总数相差甚远,所以一般分类方法的误差都会比较大。图 12.18(a)为 DBN 的分类结果图,图 12.18(b)为本节提出的算法的实验结果图。从图中可以看到本节提出的算法在很大程度上优于深度信念网络 DBN。图中蓝色的区域为水,粉色区域为植物 A 等,后者比前者分类的效果更好一些,而且几乎没有太多的分类错误,但是后者在橘色区域也就是油菜地和土黄色区域裸地分类的效果没有前者好,而且后者从视觉效果来说,分类效果是十分理想的。从正确率上而言,本节提出的算法在经过多次实验得到的正确率均值为 0.946,而原算法的正确率仅为 0.926,从总体而言本文提出的算法要优于原算法。在时间上,本节提出的算法时间为 3876.3s,原算法的时间为 5790.6s,在一定程度

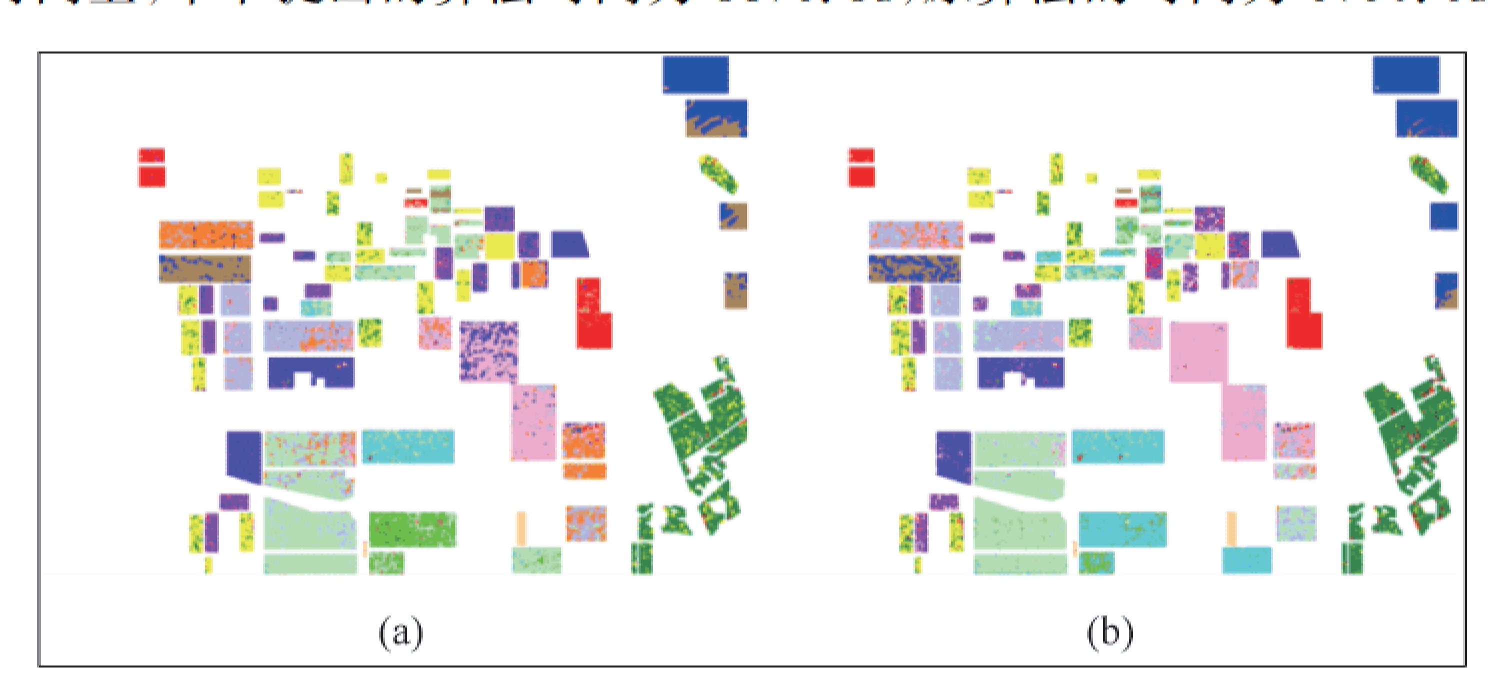


图 12.18　深度信念网络 DBN 与本章算法对 Flevoland 农田数据的实验结果图

上也加快了分类的速率。表 12.10 是两种算法在 Flevoland 农田数据上的结果显示，可以看出本节提出的算法在大多数的类别上是优于原算法的。但是本节提出的算法对有些类别分类的结果特别的差，对有些类别分类的效果特好，使得分类的结果走向两个极端，因此存在很大的不稳定性。

表 12.10 深度信念网络与本章算法在 Flevoland 农田数据上的正确率

种　类	深度信念网络 DBN	本章算法
AA	0.918	0.932
OA	0.926	0.946

2）Germany 数据实验的结果

实验随机选取了每类 10000 个样本作为训练样本，剩余的样本均为测试样本。此幅图像分辨率较上幅图像有明显提高，类别数只有三类，但是从原图可以看出本图的边缘信息特别多，而且边缘比较圆，不利于分类的进行。图 12.19(a)为深度信念网络 DBN 的分类结果图，图 12.19(b)为本节提出的算法的实验结果图。从图中可以看到本节提出的算法在很大程度上优于深度信念网络 DBN，而且红色区域可以看出本节算法的分错率更低一些。表 12.11 是两种算法在 Germany 数据上的正确率。从正确率上而言，本节提出的算法在经过多次实验得到的正确率均值为 0.7465，而原算法的正确率仅为 0.7235，从总体而言本文提出的算法要优于原算法，再从各个类别的正确率来看，仅有第一类 Built-up areas 前者分类较好，后两类均是本节算法较好。在时间上，本文提出的算法时间为 40912.4s，原算法的时间为 5475.4s，在一定程度上也加快了分类的速率。



(a) (b)

图 12.19 深度信念网络 DBN 与本章算法对 Germany 算法的实验结果图

表 12.11 深度信念网络与本章算法在 Germany 数据上的正确率

统 计 量	深度信念网络 DBN	本 章 算 法
AA	0.684667	0.694333
OA	0.7235	0.7465

3）San Francisco 数据的实验结果

实验随机选取了每类 10000 个样本作为训练样本，剩余的样本均为测试样本。此幅图像的分辨率较高，类别数为五类，对图像特征提取要求得更多。图 12.20(a)为深度信念网络 DBN 的分类结果图，图 12.20(b)为本节提出的算法的实验结果图。从图中可以看到本节提出的算法在很大程度上优于深度信念网络 DBN，尤其是蓝色海洋区域，前者有太多的散点噪声。如表 12.12 所示，从正确率上而言，本节提出的算法在经过多次实验得到的正确率均值为 0.933，而原算法的正确率仅为 0.839，从总体而言本节提出的算法要优于原算法。从各个类别上来看，有四个类别的正确率是高于原算法的，而且最后一类原算法几乎没有分出来。在时间上，本节提出的算法时间为 7098.5s，原算法的时间为 9870.6s，在一定程度上也加快了分类的速率。



图 12.20　深度信念网络 DBN 与本章算法对 San Francisco 数据的实验结果图

表 12.12　深度信念网络与本章算法在 San Francisco 数据上的正确率

统 计 量	深度信念网络 DBN	本 章 算 法
AA	0.807	0.833
OA	0.839	0.933

本节首先介绍了深度学习的快速发展以及其被广泛应用的原因，其次介绍了极化 SAR 在分类上所遇到的一些难题以及使用深度学习解决极化 SAR 的难题后出现的新的问题，及数据量过大，会影响分类的速率的问题。因此本节提出引入稀疏性的方法就可以更好地解决运算速率过低的问题。文章中也介绍了极化 SAR 数据的特征提取方法，较大多数极化 SAR 特征的提取而言，本节的方法比较简单而且节省了大量的时间，还保留了极化 SAR 的

所有原始特征,使其在后续的应用中能保证信息的完整性。最后就是构建网络模型,网络的学习以及实验的结果。可以得出,本节的方法在一定程度上可以很好地分类极化 SAR 数据,并且分类时间会大大缩短,分类精度也有所提升。

12.3.2 基于深度 PCA 网络的极化 SAR 影像地物分类

1. 模型介绍

在极化 SAR 影像地物分类领域中,提出了一种简单的基于深度 PCA 网络的极化 SAR 影像地物分类方法,该深度 PCA 网络主要由以下部分组成:级联的主成分分析(PCA),二值化与直方图统计。深度 PCA 网络中,PCA 用来学习多层的级联滤波器。单层深度 PCA 网络的训练方法可被描述为:

(1) 由于分别对极化 SAR 影像中的每个像素进行分类,将每个像素的输入数据转换为元胞数组 $\boldsymbol{I}_i \in \mathbb{R}^{m\times n}$,其中 i 表示第 i 个训练样本。分别对元胞数组 $\boldsymbol{I}_i$ 中的每一个元素施行非零填充的重叠取块 $x_{i,1}, x_{i,2}, \cdots, x_{i,mn} \in \mathbb{R}^{k_1\times k_2}$,其中块大小为 $k_1\times k_2$,元胞数组 $\boldsymbol{I}_i$ 中的第 j 个矢量块用 $x_{i,j}$ 表示。随后,通过矢量块的均值移除操作,获得 $\overline{X}_i=[\bar{x}_{i,1}, \bar{x}_{i,2}, \cdots, \bar{x}_{i,mn}]$。对所有训练样本执行相同的步骤,最终得到 $\boldsymbol{X}=[\overline{X}_1, \overline{X}_2, \cdots, \overline{X}_N]\in \mathbb{R}^{k_1k_2\times Nmn}$,训练样本总数设置成 N。

(2) 最小化重构误差,提取单层的 PCA 滤波器,则有:

$$\min_{\boldsymbol{V}\in \mathbb{R}^{(k_1\cdot k_2)\times L_1}} \|\boldsymbol{X}-\boldsymbol{V}\cdot\boldsymbol{V}^{\mathrm{T}}\cdot\boldsymbol{X}\|_F^2, \quad \text{s.t.}\ \boldsymbol{V}\cdot\boldsymbol{V}^{\mathrm{T}}=\boldsymbol{I}_L \tag{12.39}$$

其中 L 代表单层 PCA 滤波器数量,$\boldsymbol{I}_L$ 为大小是 $L\times L$ 的单位矩阵,式(12.39)的解即 $\boldsymbol{X}\cdot\boldsymbol{X}^{\mathrm{T}}$ 的前 L 个主特征向量。由此可见,深度 PCA 网络单层的 PCA 滤波器可被描述为:

$$\boldsymbol{W}_l^1=\mathrm{mat}_{k_1,k_2}(q_l(\boldsymbol{X}\cdot\boldsymbol{X}^{\mathrm{T}}))\in \mathbb{R}^{k_1\times k_2}, l=1,2,\cdots,L \tag{12.40}$$

其中函数$\mathrm{mat}_{k_1,k_2}(\boldsymbol{v})$负责将向量 $\boldsymbol{v}\in \mathbb{R}^{k_1k_2}$ 映射到矩阵 $\boldsymbol{W}\in \mathbb{R}^{k_1\times k_2}$,$q_l(\boldsymbol{X}\cdot\boldsymbol{X}^{\mathrm{T}})$负责求取 $\boldsymbol{X}\cdot\boldsymbol{X}^{\mathrm{T}}$ 的前 l 个主特征向量。

(3) 计算输出,即此层的第 l 个滤波器与输入进行卷积后的结果:

$$\boldsymbol{I}_i^l=\boldsymbol{I}_i * \boldsymbol{W}_l^1, \quad i=1,2,\cdots,N \tag{12.41}$$

其中 N 为训练样本的总量,$*$ 表示 2D 卷积。将此层滤波器卷积的结果,即输出,用作深度 PCA 网络下一层的输入数据。

深度 PCA 网络除最后一个训练层外,其他每层仅执行卷积滤波器层和非线性处理层。最后一个训练层的特征池化层处理策略为:

(1) 对卷积结果做二值化操作。

利用函数$\{H(\boldsymbol{I}_i^l * \boldsymbol{W}_l^2)\}_{l=1}^{L}$对深度 PCA 网络最后一个训练层的滤波器卷积得到的输出结果做二值化处理,其中 $H(\cdot)$为单位阶跃函数,正数作为输入则函数输出为 1,否则,函数输出为 0;

（2）二值化结果执行十进制数值化操作。

将二值化后的输出看作一组 L 位二进制向量，将该组 L 位二进制向量转换成一个十进制数值：

$$T_i^l = \sum_{l=1}^{L} 2^{l-1} H(\boldsymbol{I}_i^l * \boldsymbol{W}_l^2) \tag{12.42}$$

（3）利用块直方图对十进制化结果进行统计。

本节中对得到的对应的十进制数值的输出图像 $T_i^l, l=1,2,\cdots,L_1$，进行重叠取块，依据实验数据设定块的大小和直方图取块的步长，通过直方图统计对全部子块中的十进制值进行计算，若块的个数是 B，组合 B 个直方图块对应的十进制化结果，构成一个向量 $B\text{hist}(T_i^l)$。通过深度 PCA 网络的训练，$\boldsymbol{I}_i$ 所对应的特征集合可以通过向量 $B\text{hist}(T_i^l)$ 得到表征。下面的算法给出了本节算法的具体实现策略：

算法 12.1

基于深度 PCA 网络的极化 SAR 影像地物分类算法
1. 采用精致极化 LEE 滤波法 3×3 窗口滤波，按照第 3.3.1 节中的方法获得深度 PCA 网络的输入数据；
2. 选取部分有标记数据用来训练，其余的用作测试；
3. 训练第一层深度 PCA 网络；
4. 训练第二层深度 PCA 网络；
5. 对第二层的训练结果进行特征池化；
6. 训练 SVM 分类器；
7. 对未标记样本进行预测分类；
8. 计算准确率，显示结果。

2. 实验结果与分析

1）Germany 地区实验设置和实验结果分析

在这一小节中，我们选用 1989 年获取的荷兰 Flevoland 地区的 L 波段的地物数据来进行实验，训练样本每类选取 1000 个，共 15 类。首先，利用精致 Lee 滤波法处理原始数据，然后，将 Flevoland 影像的每一个极化 SAR 像素作为一个处理单位，通过从协方差矩阵 $\boldsymbol{C}$ 中提取出 9 个独立的元素，并与数据分布特征参数 α、散射特征和偏振特征进行组合归一化，作为深度 PCA 网络的原始输入数据。其中图 12.21(d)为利用该算法所获得的分类结果图，图 12.21(a)～图 12.21(c)分别为针对 PCA、SVM 和 Wishart 三种对比方法的分类结果图。通过图 12.21 和表 12.13 可以明显地看出，本节所提出的算法能够准确有效地区分各类地物数据，少数类别的地物分类正确率可以到达 100%，且所有类别的地物分类正确率均高于 95%。无论是从整体分类精度还是从单个类别的分类精度考虑，本节算法均优于其他三种方法，大大提高了分类效果。

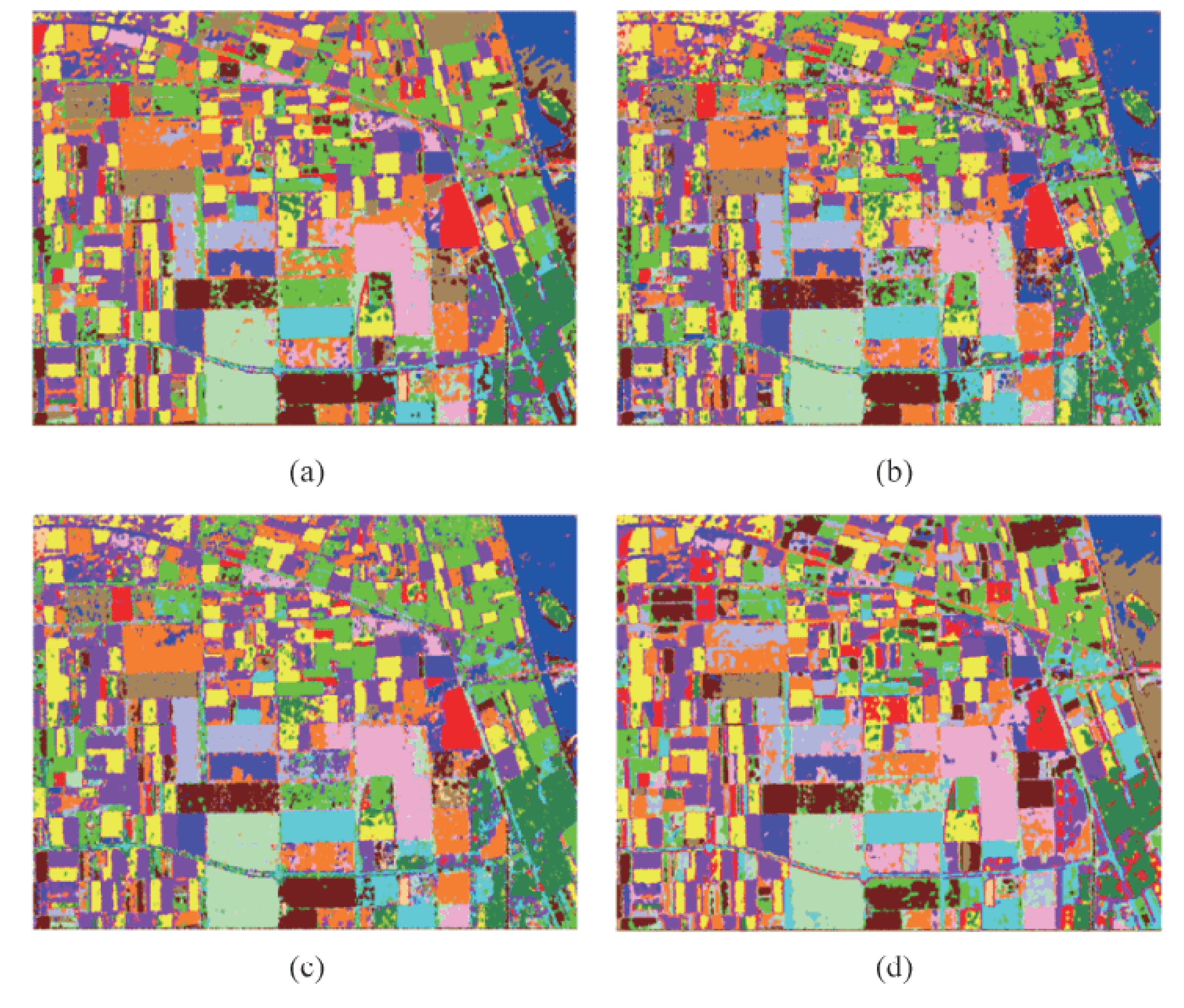


图 12.21　Flevoland 影像分类结果图

(a) PCA 分类结果图；(b) SVM 分类结果图；(c) Wishart 分类结果图；(d) 深度 PCA 网络分类结果图

表 12.13　几种对比方法对 1991 年 Flevoland 地区分类准确度对比

统计量＼方法	PCA	SVM	Wishart	Proposed
AA	0.9359	0.9577	0.9045	0.9981
OA	0.9018	0.8728	0.9464	0.9864

2）Germany 地区实验设置和实验结果分析

对于 ESAR 获取的 Oberpfaffenhofen，Germany 地区四视的全极化数据集，大小为 1300×1200，主要包括了三类：城区、林区和开发区域影像数据，每类选取 5000 个训练样本，图 12.22(d)展示了本章算法的分类效果图，图 12.22(a)为利用传统的主成分分析 PCA 和 SVM 直接进行分类的结果，图 12.22(b)是直接利用 SVM 进行分类的分类结果，图 12.22(c)给出了利用 Wishart 分类结果图，表 12.23 给出了本章算法与三种对比实验方法的正确率统计。从图 12.22(a)～图 12.22(c)和表 12.14 中可以看出，相比其他几种对比方法(SVM、PCA 和 Wilshart 分类方法)，本节算法识别率分别高出 10.68%、9.98%和 12.36%。SVM 和 PCA 方法城区部分杂点过多，接近一半错分为了林区，本节算法分类结果图无论是从视觉效果还是从分类的正确率统计进行分析，两者均兼有明显的优势和提高。

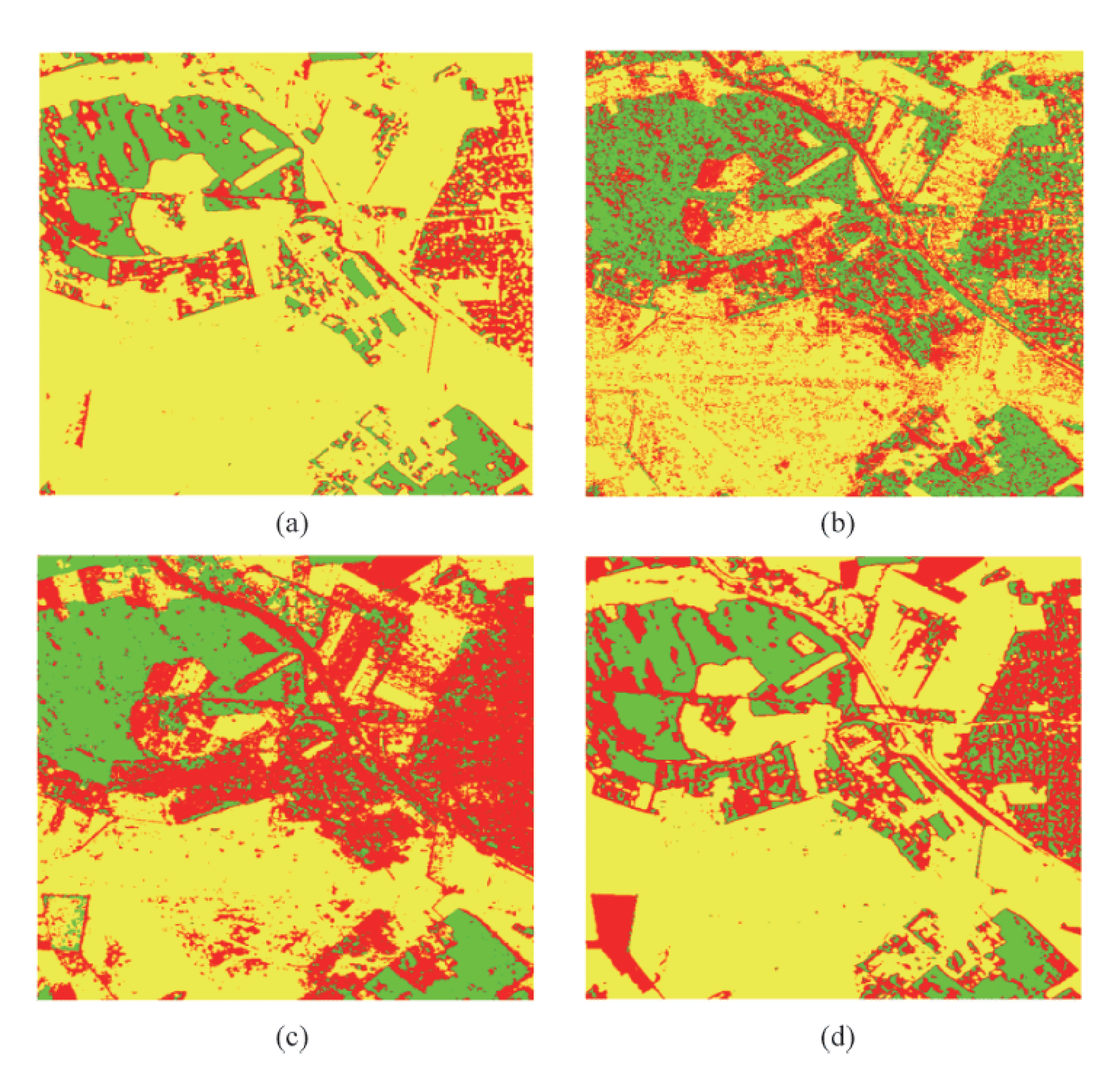


图 12.22　Germany 影像的分类结果对比图

(a) 传统的 RPCA 分类结果图；(b) SVM 分类结果图；(c) Wishart 分类结果图；(d) 深度 RPCA 网络分类结果图

表 12.14　几种对比方法对 Germany 影像分类精度比较

统计量 \ 方法	PCA	SVM	Wishart	Proposed
AA	0.5778	0.4001	0.6063	0.8236
OA	0.6868	0.6170	0.6351	0.8725

3）SF Bay 地区实验设置和实验结果分析

极化 SAR 影像地物数据为 2008 年获取的大小约 1380×18000 的旧金山海湾 San Francisco 影像，训练样本每类选取 3000 个，共 5 类，图 12.23 给出了利用本节算法的分类效果图，图 12.23(c)给出了利用传统 PCA 直接进行分类的结果，图 12.23(d)是直接利用 SVM 进行分类的分类结果，图 12.23(c)给出了利用 Wishart 分类的结果图，表 12.15 为本节算法与其他三种对比实验方法的正确率统计。从图 12.23(a)～图 12.23(c)和表 12.15 中可以看出，传统 PCA 对于低密度城区和高密度城区的错分、误分很明显，利用 Wishart 分类方法对高密度城区和开发区很不理想，整体分类情况较差，SVM 整体情形相对较好。相比上述三种对比方法，本节算法通过深度 PCA 网络的构建，可以有效地学习此影像的特征，分类的正确率也有了更明显的提高。

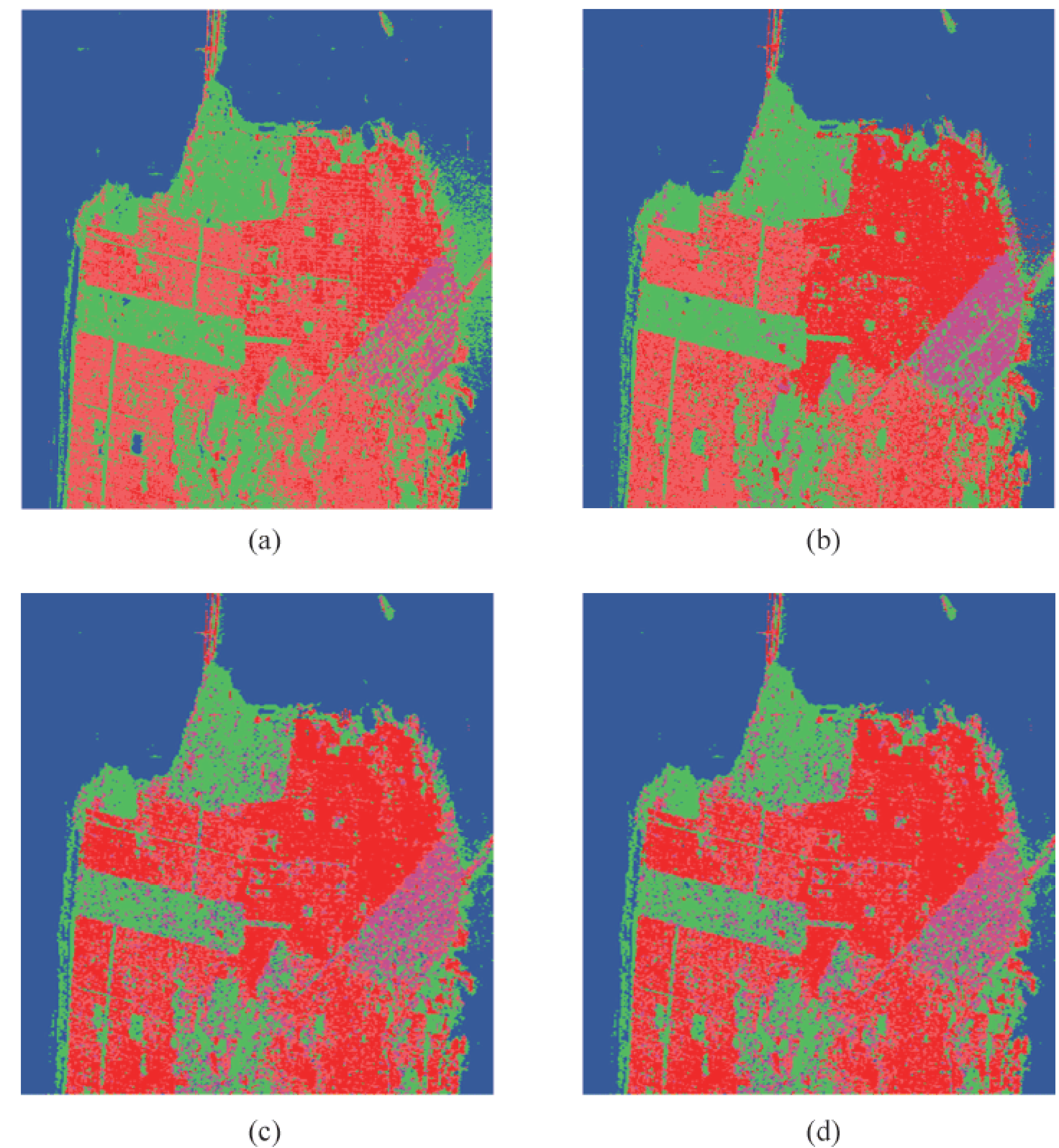


图 12.23 算法 12.1 和三种对比方法对旧金山地区的分类结果对比图

(a) 传统的 RPCA 分类结果图；(b) SVM 分类结果图；(c) Wishart 分类结果图；(d) 深度 RPCA 网络分类结果图

表 12.15 几种对比方法对旧金山海湾影像分类精度比较

方法 统计量	PCA	SVM	Wishart	Proposed
AA	0.7485	0.8844	0.8817	0.8944
OA	0.7809	0.8610	0.8011	0.8959

12.4 基于第二代深度神经网络的 PolSAR 影像地物分类

12.4.1 基于深度复卷积网络的 PolSAR 影像地物分类

1. 模型介绍

极化 SAR(多极化 SAR、全极化 SAR)是一种多通道相干微波成像系统，是单极化 SAR 的扩展系统。它通过矢量测量方法来获取地物目标信息。众所周知，电磁波是一种矢

量波，包含幅度、相位、频率等分量。

我们获取的极化 SAR 数据多为基于测量数据的极化相干矩阵 $\boldsymbol{T}$。传统的特征提取方法为分别取极化相干矩阵 $\boldsymbol{T}$ 的实部、虚部，以及对矩阵元素进行取模操作，将得到的图像特征用于极化 SAR 图像分类。

$$\boldsymbol{T}=\begin{bmatrix} a & c-id & h+ig \\ c+id & b & e+if \\ h-ig & e-if & l \end{bmatrix} \longrightarrow a,b,c,d,e,f,g,h,l,|c-id|,|h+ig|,|e+if| \tag{12.43}$$

这种特征提取方法没有考虑到极化 SAR 图像的相位信息，因而对背景复杂的极化 SAR 图像难以取得较高的分类精度。据此，我们考虑将卷积神经网络延拓至复数域进行运算，直接处理复数数据，充分利用极化 SAR 数据的方向信息，增强模型的泛化能力。卷积神经网络中的核心模块卷积流（卷积、池化、非线性、批量归一化）运算规则改进如下：

（1）复数域卷积：当输入数据为复数形式，即 $\boldsymbol{x}=\boldsymbol{a}+\mathrm{j}\cdot\boldsymbol{b}\in\mathbb{C}^{n\times m}$，那么卷积核最简单的方式为 $\boldsymbol{w}=\boldsymbol{u}+\mathrm{j}\cdot\boldsymbol{v}\in\mathbb{C}^{u\times v}$，所以有 $\boldsymbol{x}$ 与 $\boldsymbol{w}$ 的卷积为

$$\boldsymbol{x}*\boldsymbol{w}=(\boldsymbol{a}*\boldsymbol{u}-\boldsymbol{b}*\boldsymbol{v})+\mathrm{j}\cdot(\boldsymbol{a}*\boldsymbol{v}+\boldsymbol{b}*\boldsymbol{u})\in\mathbb{C}^{(n-u+1)\times(m-v+1)} \tag{12.44}$$

另外偏置设为 $c=\alpha+\mathrm{j}\cdot\beta\in\mathbb{C}^{1\times1}$。

（2）复数域非线性：假设复数域卷积操作完成后的输出为 $\boldsymbol{\Gamma}=\boldsymbol{x}*\boldsymbol{w}+c$，那么非线性函数 φ 与之前实数域上的取法一致，但这里的操作需分为实部与虚部，即

$$\varphi(\boldsymbol{\Gamma})=\varphi(\mathrm{Re}(\boldsymbol{\Gamma}))+\mathrm{j}\cdot\varphi(\mathrm{Im}(\boldsymbol{\Gamma}))\in\mathbb{C}^{(n-u+1)\times(m-v+1)} \tag{12.45}$$

（3）复数域池化：假设卷积非线性处理完后的输出为 $\boldsymbol{\Omega}=\varphi(\boldsymbol{\Gamma})$，也与之前的池化方式一样，但需注意的仍是分为实部与虚部操作，即

$$\boldsymbol{P}=\mathrm{Maxpooling}(\mathrm{Re}(\boldsymbol{\Omega}),r)+\mathrm{j}\cdot\mathrm{Maxpooling}(\mathrm{Im}(\boldsymbol{\Omega}),r)\in\mathbb{C}^{\frac{(n-u+1)}{2}\times\frac{(m-v+1)}{2}} \tag{12.46}$$

其中的 r 为池化半径。

（4）复数域批量归一化：与之前的归一化方式一样，对 P 进行实部与虚部归一化，记为

$$\boldsymbol{F}=\mathrm{Normalization}(\mathrm{Re}(\boldsymbol{P}))+\mathrm{j}\cdot\mathrm{Normalization}(\mathrm{Im}(\boldsymbol{P})) \tag{12.47}$$

（5）复数域全连接层：得到复数域批量化处理的特征映射 $\boldsymbol{F}\in\mathbb{C}^{T@n_S\times m_S}$，这里的 S 为卷积流模块个数，T 为特征映射图个数。将 $\boldsymbol{F}$ 向量化后得到 $\mathrm{Vector}(\boldsymbol{F})\in\mathbb{C}^{(T\cdot n_S\cdot m_S)}$，并将其映射至 $\boldsymbol{F}_S\in\mathbb{C}^{K\times1}$。

当我们将卷积流延拓至复数域后，不失一般性，得到的深层“复”特征映射为 $\boldsymbol{F}_S$。由于类标不存在复数形式，所以为了与输出类标对应，那么已知 $\boldsymbol{F}_S$ 后，最简单的方式为“复”分类器的设计，即

$$\mathrm{Predict}(y=k\mid\boldsymbol{F}_S(x),\omega_k)=\frac{1}{Z}\exp(\boldsymbol{F}_S\odot\omega_k) \tag{12.48}$$

其中“$\odot$”为直乘，Z 为和项，利用负对数信息熵建构损失函数。

待分类的 PolSAR 图像选用 NASA/JPL 实验室 AIRSAR 系统的 L 波段荷兰 Flevoland 地区的全极化数据，图像大小为 750×1024，15 类地物。

2. 实验结果与分析

硬件平台为：Intel(R) Xeon(R) CPU E5-2630，2.40GHz×16，内存为 64GB。

软件平台为：MxNet。

在上述仿真条件下进行实验，即分别从 PolSAR 数据的每个类别中随机选取 200 个有标记的像素点作为训练样本，其余有标记的像素点作为测试样本，得到如图 12.24 所示的分类结果。

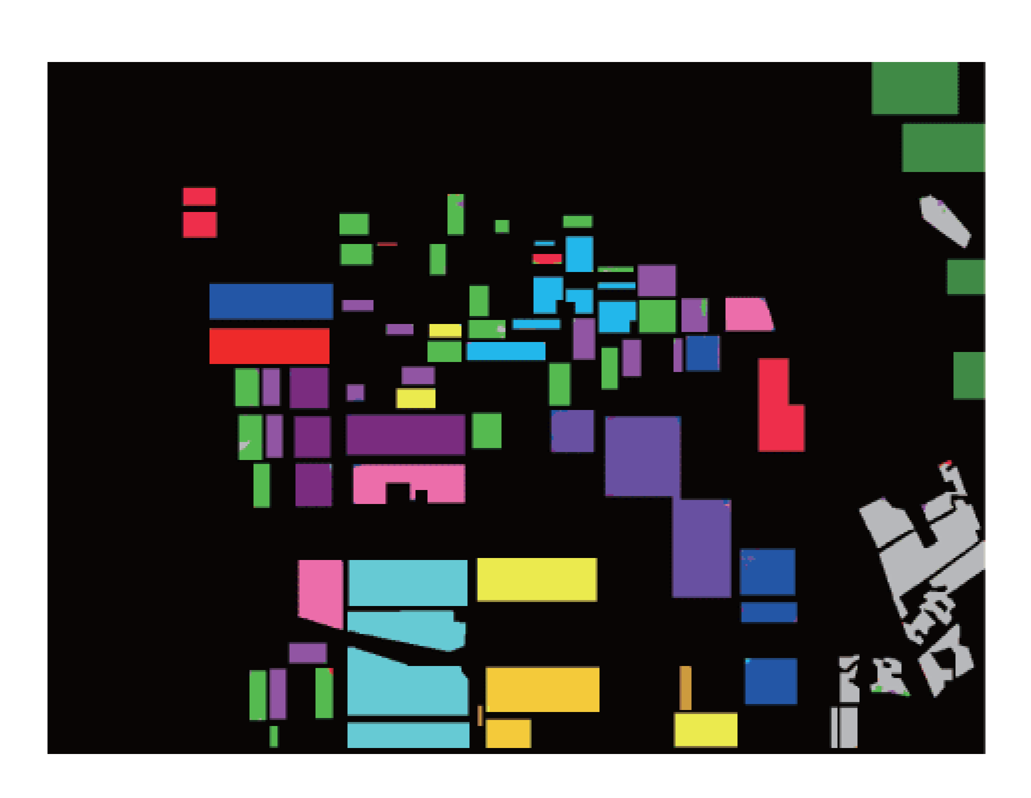


图 12.24 基于深度复卷积网络的分类结果图

从图 12.24 中可以看出：分类结果的边缘非常清晰且区域一致性较好，表明所提出的深度复卷积网络适用于处理 PolSAR 图像分类问题。

深度复卷积网络与深度卷积神经网络在测试数据集上的各类分类精度对比如表 12.16 所示。

表 12.16 各类分类精度对比

类　别	卷积神经网络	深度复卷积网络
Stembeans	97.74%	99.75%
Rapeseed	91.49%	92.11%
Bare soil	100%	100%
Potatoes	97.23%	99.12%
Wheat	92.36%	98.61%
Wheat 2	100%	100%
Peas	99.25%	99.37%
Wheat 3	99.75%	100%
Lucerne	99.50%	100%
Barley	99.75%	99.50%

续表

类　　别	卷积神经网络	深度复卷积网络
Grasses	98.86%	99.75%
Beet	98.61%	98.74%
Buildings	97.98%	98.61%
Water	100%	100%
Forest	99.62%	99.62%

由表 12.16 可看出，深度复卷积网络较之卷积神经网络在大部分地物上都有优势。且在 Stembeans、Potatoes、Lucerne、Grasses、Buildings 等类别上分类精度提升较大。

再依次减少训练样本，从每类中选取 100 个、50 个有标记的像素点作为训练样本，将深度复卷积网络与卷积神经网络的测试数据集分类精度进行对比，结果如表 12.17 所示。

表 12.17　不同数目训练样本下，测试数据集分类精度对比

每类训练样本数目	训练样本所占比例	卷积神经网络	深度复卷积网络
200	1.8%	99.00%	99.41%
100	0.9%	97.60%	97.94%
50	0.5%	95.33%	96.37%

从表 12.17 可见，训练样本占样本总数的 1.8%、0.9%、0.5%时，深度复卷积网络的测试数据集分类精度均明显高于卷积神经网络。在训练样本数目较少的情况下，优势明显。

综上，通过将卷积神经网络延拓至复数域进行运算，有效提高了图像特征的表达能力，PolSAR 图像的分类精度得到显著提升。

12.4.2　基于生成式对抗网的 PolSAR 影像地物分类

1. 模型介绍

深度卷积生成式对抗网(Deep Convolution GAN，原文缩写为 DCGAN，现缩写为 DAN-Convolutional)是 2015 年 Radford A 等在 Computer Science 上发表的论文 *Unsupervised representation learning with deep convolutional generative adversarial networks* 中提出的，在 TensorFlow 平台上有相关实现代码。DAN 结合有监督学习的 CNN 和无监督学习的 GAN，能够进行无监督表征学习，训练好的生成器和判别器的隐含层都可以对图像进行特征表示，重用其训练好的生成模型和判别模型，能够应用于图像分类任务。

DAN 的结构是在原始 GAN 的基础上，将生成器和判别器的隐含层全部用卷积层实现。虽然 GAN 本身训练不需要特定的启发式损失函数，优化过程是一个“二元极大极小博弈”问题，但是 GAN 本身训练十分不稳定。作者根据自己在 CNN 领域的工程经验，提出和评估了一系列约束使得网络在训练中稳定。

其模型的方法和核心主要有：

(1) 用生成器的带步长卷积(Strided Convolutions)替换所有池化层;

(2) 用判别器的微步幅卷积(Fractional Strided Convolutions)替换所有池化层;

(3) 在生成器和判别器上都使用批标准化(Batchnorm),这个策略能有效地解决初始化不当引起训练崩溃的问题,但如果将批标准化应用于所有层又会引起模型的不稳定,所以采取的措施为在生成器的输出层和判别器的输入不使用批标准化;

(4) 删除深度网络中的全连接层,论文中提到,原始 CNN 中一般使用的是全局池化(global pooling),这样虽然可以增加模型的稳定性,但是网络的收敛速度会降低;

(5) 生成器中输出层用 Tanh 激活函数,其他所有层用 Relu 激活函数;

(6) 判别器中所有层的激活函数都用 LeakyRelu。

具体网络模型如图 12.25 所示。

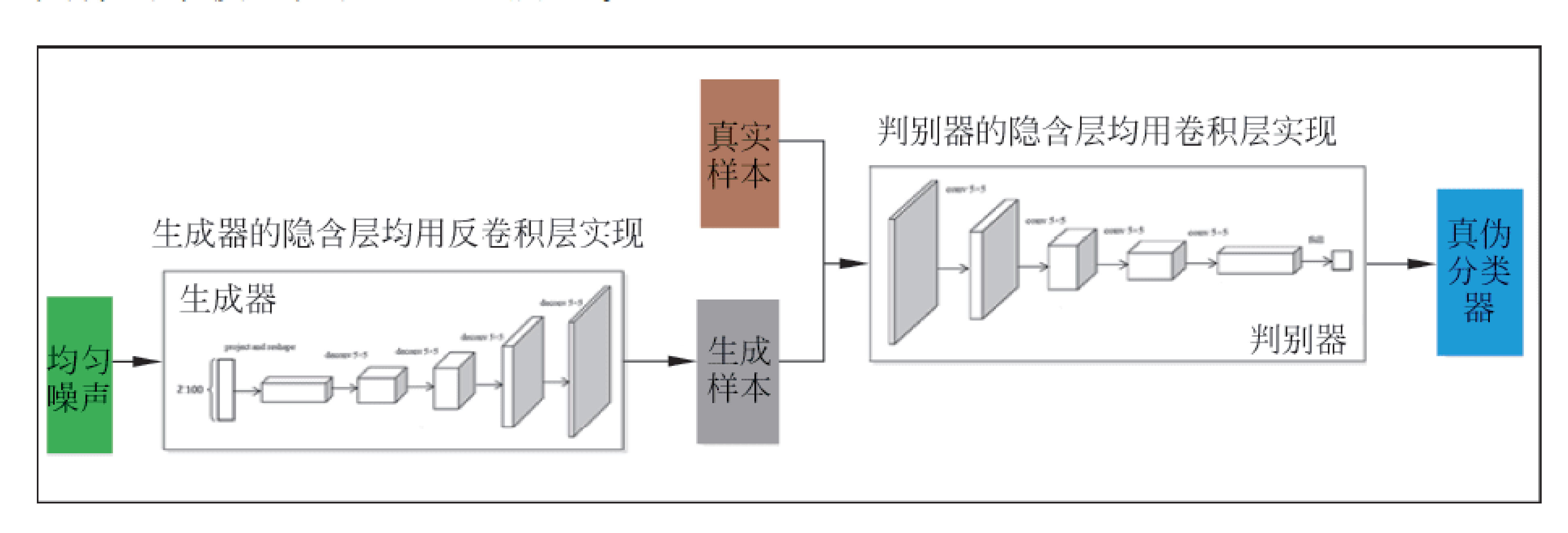


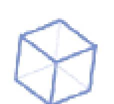
图 12.25　DAN 网络结构

对于生成器,输入为 100 维的均匀噪声,第一层为全连接层,将 100 维的向量投影成 4×4 大小的 feature map,通道数为 512。然后依次用四层步长为 5×5 的带步长卷积,这样使得每次卷积后图像尺寸加倍,通道数减半。最后转换为 64×64 大小的 RGB 三通道图片,这些图片就是生成的假样本。

对于判别器,输入为真实的样本和生成器生成的假样本,均为 64×64 大小的 RGB 三通道图片。判别器与生成器的各层的图像尺寸和通道数保持一致。判别器的前四层依次图像尺寸减半,通道数加倍,生成高级特征表示。最后一层为一个 logistics 回归二分类器,输出为一个标量,即对样本真实性的评分,表示是否为真实样本。

训练中超参数设置如下:采用 mini-batch 进行训练,训练的 batchsize 为 64;以往的 GAN 都采用 momentum 优化器加速训练,DAN 采用 adam 优化器进行学习训练,且学习率为 0.0002;所有的参数初始化都是从正态分布得到的,正态分布设置的参数为均值为 0,方差为 0.02;设置 LeakyReLU 的斜率为 0.2;将 momentum 参数 beta 从 0.9 降为 0.5,有助于训练稳定,防止震荡。

基于 DAN 的半监督极化 SAR 图像分类。我们为充分利用 DAN 网络具有的无监督学习表征的特性,并将其引用到极化 SAR 图像分类领域,提出一种自动进行特征学习而后对极化 SAR 图像分类的方法。这里给出基于 DAN 的半监督极化 SAR 图像分类流程图,如图 12.26 所示。



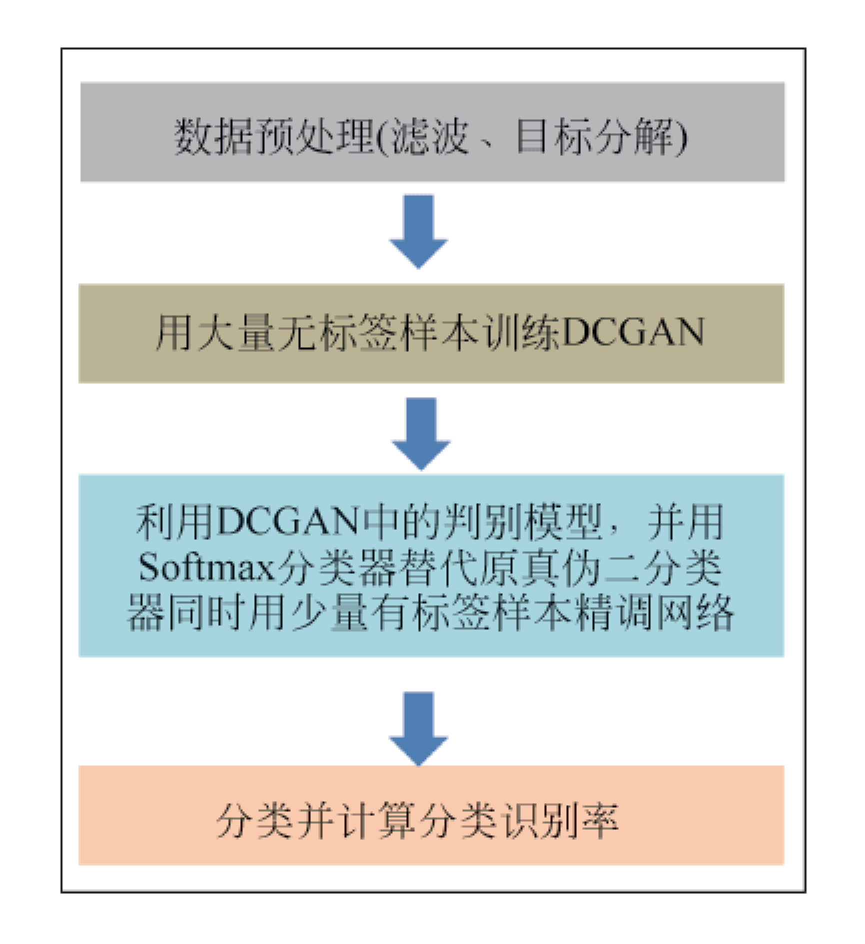


图 12.26　基于 DAN 的半监督极化 SAR 图像分类流程图

具体分类算法步骤为：

(1) 首先对极化 SAR 图像采用 Pauli 分解，分解为几个极化特征并形成伪彩图代表原极化 SAR 数据。将 pauli 分解后的数据切成 $64\times64\times3$ 的块，用这个块代表第(32,32)像素点。

(2) 然后用极化 SAR 图像中无标签样本对训练网络，即 DAN 进行训练。训练网络模型如图 12.27 中所示，分为生成网络 G 和判别网络 D。生成器 G 的输入为服从均匀分布的噪声矢量，这是传统 GAN 的生成器唯一输入。生成器 G 的输出为极化 SAR 数据，与判别器 D 的输入形式相同。极化 SAR 图像无标签样本的极化特征作为判别器 D 的输入，而判别器的输出为一个标量，即一个二分类器，输出是否为真实的训练样本。

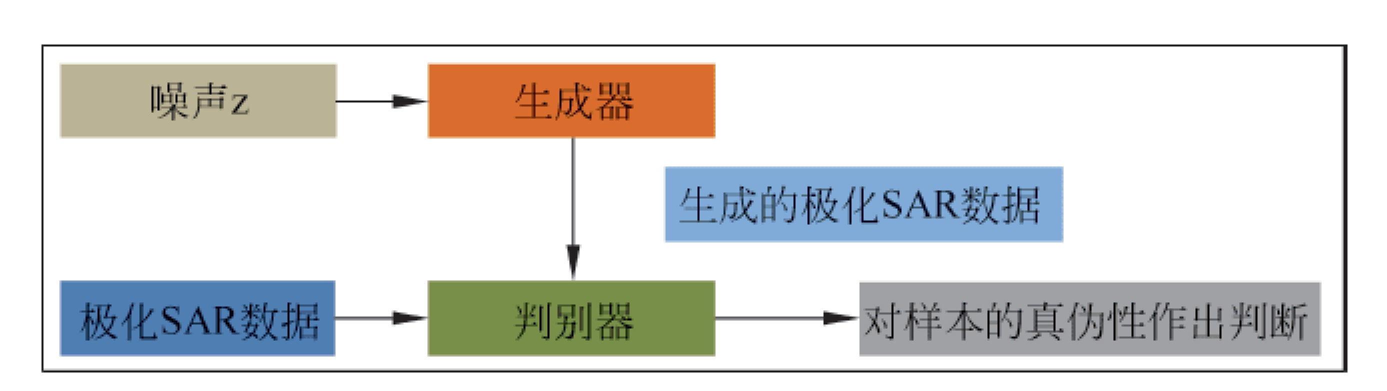


图 12.27　训练网络模型

(3) 利用训练好的 DAN 中的判别器 D，将二分类器更换为 softmax 分类器，构造分类网络模型，如图 12.28 所示。然后用少量有标记样本训练分类器。训练好分类器以后，再接着用有标签样本对整个分类网络进行精调。最后，可以输入测试集用分类网络对其进行分类。

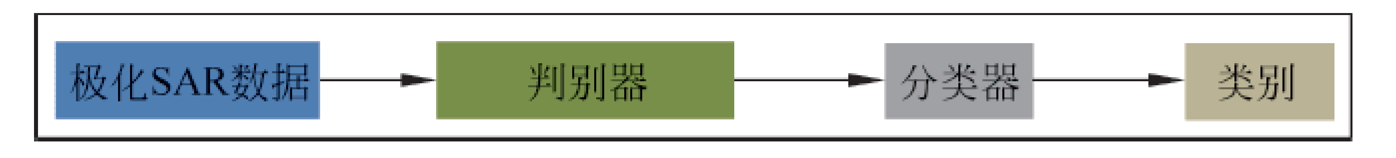


图 12.28　分类网络模型

2. 实验结果与分析

1）仿真条件

硬件平台：HP Z840

深度学习平台：TensorFlow

2）仿真内容

实验选取旧金山海湾影像，总像素点个数为1800×1380。首先在所有像素点中选取86 940个(占总像素点的3.5%)无标签像素点，记录这些点的位置。然后在这些点周围取64×64的块组成无标签训练样本，将无标签样本输入训练网络模型进行无监督训练。然后在标记过的像素点(有标签像素点共1 804 087个)中每个类别里选择0.5%的有标签像素点，记录这些点的位置，其余的有标记像素点取块作为测试集。然后在这些点周围取64×64的块组成有标签训练样本集，将有标签样本集输入分类网络模型训练。最后，将测试集输入分类网络模型进行分类，再计算分类准确率。分类结果如图12.29所示，其分类精度达99.4346%。将本方法与传统卷积神经网络CNN的测试分类精度进行比较，结果如表12.18所示，用CNN分类结果为图12.30。

表12.18　本节方法与卷积神经网络方法实验结果对比

分类方法	卷积神经网络	本　方　法
类别1(%)	99.9832	99.9913
类别2(%)	97.1476	98.5980
类别3(%)	91.1120	98.5554
类别4(%)	91.2401	99.6494
类别5(%)	95.0531	99.0883
总准确率	97.4149	99.4346

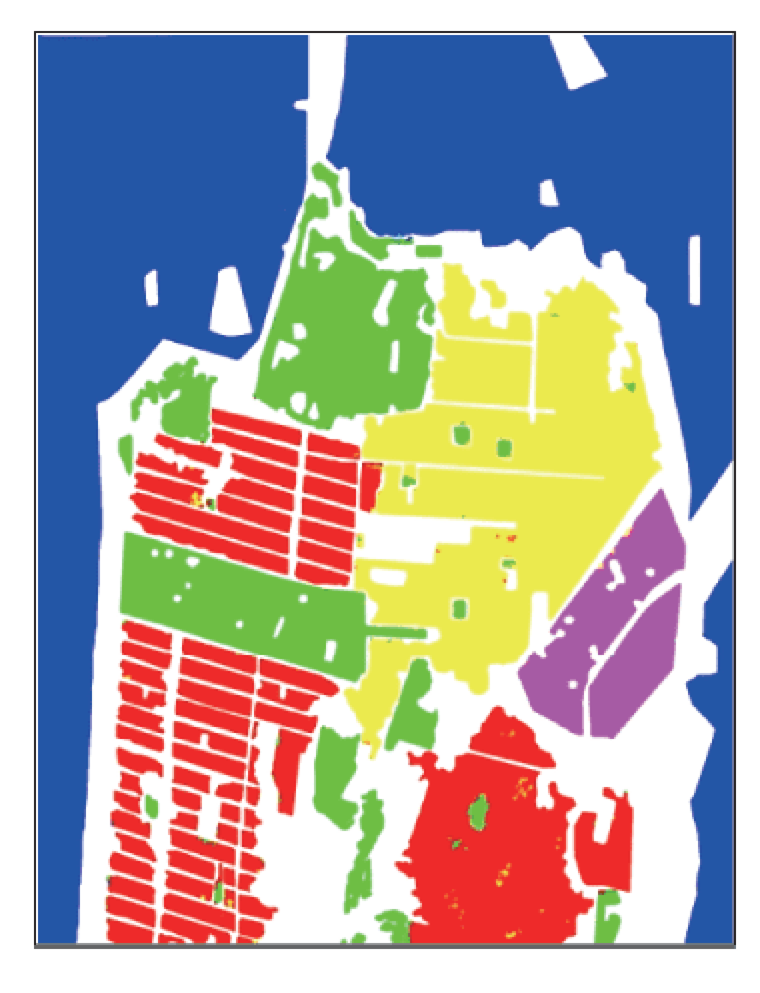


图12.29　本方法分类结果图



图12.30　CNN分类结果图

由表12.18的对比实验结果可以看出，本方法的每一类别的分类精度均比传统CNN分类效果好，提高了分类精度。而且由分类结果图可以看出，两种方法对水域的分类准确率都很高，分别为99.9913%和99.9832%。而在对农田、高密度城区和开发区的分类时，用CNN进行分类略逊于本方法。在对低密度城区分类时，CNN的错分和误分情况比较严重，CNN分类结果图中低密度城区部分的噪声杂点也比较多，分类精度方面比本方法的精度低了7.4434%。而本方法的结果图中分类结果的区域一致性较好，不同区域划分后的边缘清晰可辨，且保持了细节信息，分类结果图中噪声也比较少。不管从分类精度还是视觉效果，本方法都具有明显的优势。

这证明了本方法使用DAN进行特征提取，能够从大量无标记数据中学习数据分布特性，具有很好的特征表示能力。而且本方法相比其他极化SAR分类方法，不需要添加滤波过程，因为DAN模型本身具有降噪功能，它能自动滤除极化SAR数据中相干斑噪声，然后学习数据特征。同时，本方法通过重用DAN的判别器模型对极化SAR能生成图像表征，相比其他深度学习特征提取的方法，无须启发式损失函数，也能很好地表征图像，即使用少量的有标记样本对极化SAR数据分类仍可以达到很高的分类精度。

12.4.3　基于深度残差网络的PolSAR影像地物分类

1. 模型介绍

深度卷积神经网络在图像分类方面引发了一系列突破。通过改变叠层的数量(深度)，深度网络自然整合低/中/高水平的功能，端到端多层方式的分层器，和特征的“水平”都变得更加丰富。

1）残差学习

将$H(x)$假设为由几个堆叠层匹配的(不一定是整个网)基础映射，用x表示这些第一层的输入。假设多元非线性层能逼近复杂的函数，也就相当于假设它们可以逼近残差函数，例如$H(x)-x$(假设输入和输出在同一规模)。因此我们非常明确地让这些层近似于残差函数，而并非期待堆叠层近似于$H(x)$。所以原函数变成了：$F(x)+x$。尽管两种形式都能逼近期望函数，但学习难易度可能不同。新的构思源于反常的精准度下降问题。如果添加的层可以被构造为恒等映射，那么一个更深度模型的训练误差，不应大于与其相应的更浅的模型训练误差。精准度下降问题表明，求解器在通过多个非线性层近似于恒等映射方面有困难。随着残差学习重构，如果恒等映射是最佳的方法，那么求解器可以简单地驱动多个非线性层的权重趋向于零，以便逼近恒等映射。在现实情况中，恒等映射不可能是最优的，但我们的方法可能有助于事先处理该问题。如果最优函数与趋近于零映射相比更趋近于身份函数，那么与学习一个新函数相比，求解器更容易找到关于恒等映射的干扰。我们通过实验展示所学到的剩余函数一般有小的响应，这表明恒等映射提供了合理的预处理。

2）快捷方式的恒等映射

我们对每一个堆叠层都采用残差学习，一个构建模块如图12.31所示。正式地说，本文

构建模块定义为：

$$y = F\{x, \{W_i\}\} + x \tag{12.49}$$

其中 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 是考虑到的层的输入和输出向量。函数代表学习的残差函数。如图 12.31 所示有两个层，而且消除了简化符号的偏见. $F+\boldsymbol{x}$ 的操作是由快捷连接和增加的元素智能进行的。在增加之后我们采用了第二非线性特性。

式(12.49)中介绍的快捷连接，没有额外的参数和复杂的计算。这不仅在实践中有吸引力，它在对比平原和残差网络方面也同样重要。有着相同数量的参数、深度、宽度和计算成本时(除了可以忽略不计的元素智能的添加)，可以对平原和残差网络进行简单的对比。式(12.49)中 $\boldsymbol{x}$ 和 F 的大小必须相同。如果不同(例如改变输入和输出渠道)我们可以通过快捷连接线性投影 $\boldsymbol{W}_s$ 来匹配维度：

$$y = F\{x, \{W_i\}\} + W_s \cdot x \tag{12.50}$$

可以使用一个正方形矩阵 $\boldsymbol{W}_s$。但我们会通过实验表明，恒等映射足以用于解决精准度下降问题并且是非常合算的，因此只有在匹配维度时，才使用 $\boldsymbol{W}_s$。残差函数 F 的形式是灵活的，本文的实验涉及一个有两层或三层或者更多层的函数 F。但如果 F 仅仅只有单层，式(12.50)就类似于线性层。我们还注意到，虽然上面的符号为了简单起见是关于完全连接的层，但它们适用于卷积层。函数可以代表多个卷积层。增加的元素智能在两个特征映射上通过通道对通道的方式进行。本实验的模型如图 12.32 所示，本实验一共有 5 个残差块，共计 32 层。

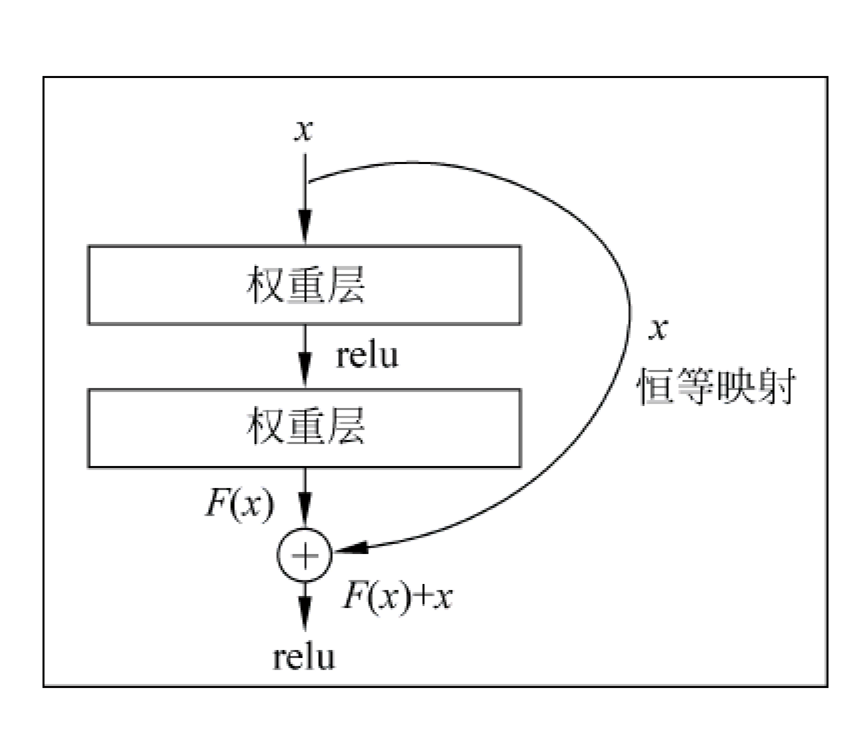


图 12.31　残差块示意图

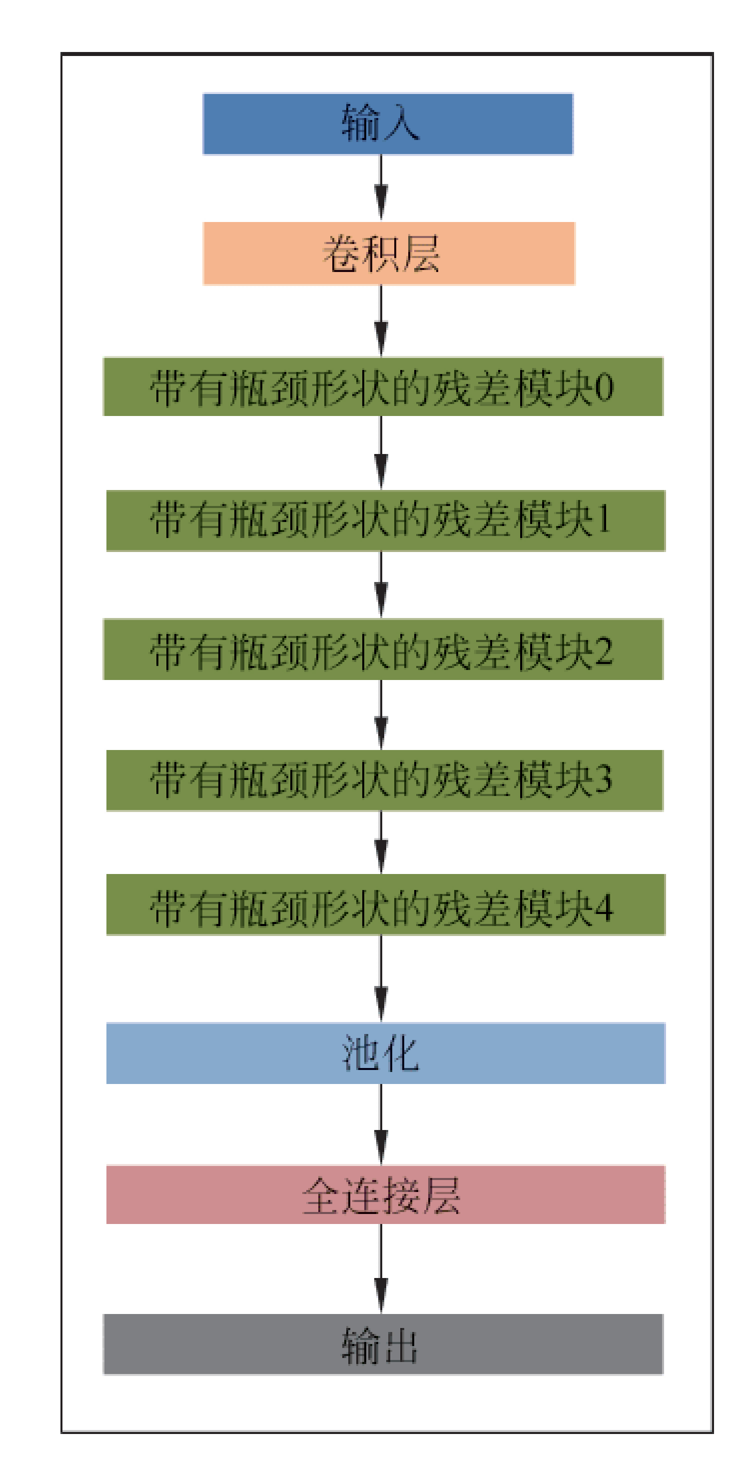


图 12.32　流程图

2. 实验

实验结果如表 12.19 所示。

表 12.19　实验结果

实　验	训练集/测试集	训练集准确率	测试集准确率
实验一	4%/96%	97.68%	96.806%
实验二	3%/97%	98.71%	95.737%

实验结果如图 12.33 和图 12.34 所示。

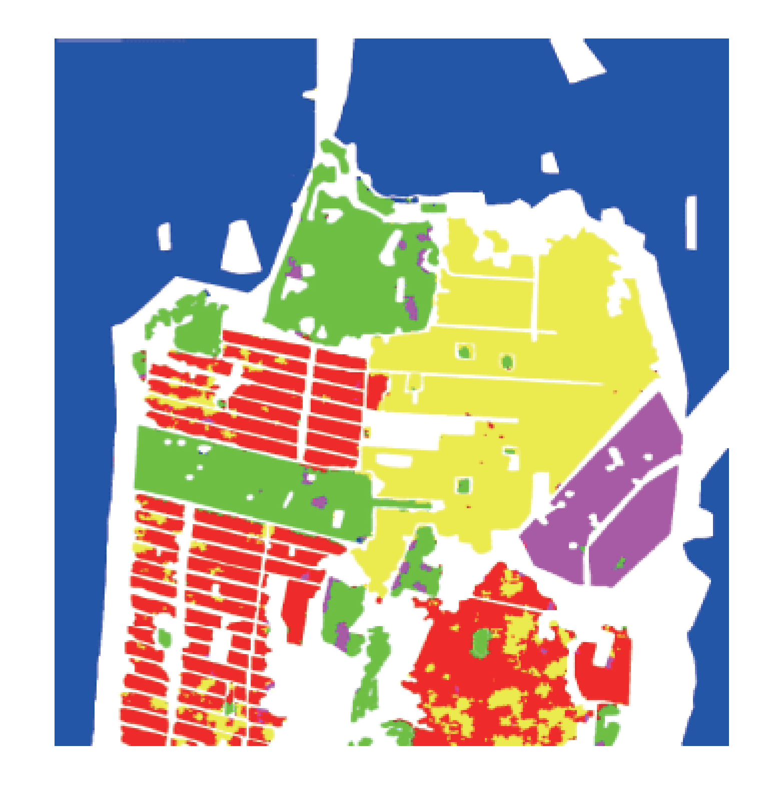


图 12.33　实验一结果图



图 12.34　实验二结果图

3. 实验总结

本节采用 1800×1380 的旧金山海湾极化 SAR 影像地物数据，图 12.33 给出了训练数据取 4%时的结果，图 12.34 给出的则是训练数据取 3%时的分类结果。通过深度残差网的构建，较为准确有效地表示出极化 SAR 旧金山海湾的地物特性和极化机理特征等，无论是从视觉效果还是从分类的正确率统计方面进行分析，两者均有良好的结果。由于极化 SAR 数据特有的极化特性、散射特性和领域信息等，成为数据训练学习的关键目的所在，深度残差网在基于数据的极化特性和散射特性的基础上，学习了极化 SAR 数据的高阶特征和低阶特征，提高了分类精度。

参考文献

[12.1]　周晓光，匡纲要，万建伟. 极化 SAR 图像分类综述[J]. 信号处理，2008，24(5)：806-812.

[12.2]　张海剑，杨文，邹同元，等. 基于四分量散射模型的多极化 SAR 图像分类[J]. 武汉大学学报信息科学版，2009，34(1)：122-125.

[12.3]　王海江. 极化 SAR 图像分类方法研究[D]. 电子科技大学，2008.

[12.4] 吴永辉，计科峰，郁文贤. 基于支持向量机的极化 SAR 图像分类[J]. 现代雷达，2007，29(6)：57-60.
[12.5] Doulgeris A P，Anfinsen S N，Eltoft T. Classification With a Non-Gaussian Model for PolSAR Data[J]. Geoscience & Remote Sensing IEEE Transactions on，2008，46(10)：2999-3009.
[12.6] Ferro-Famil L，Pottier E. Urban area remote sensing from L-band PolSAR data using Time-Frequency techniques[J]. 2007：1-6.
[12.7] Skriver H，Dall J，Ferrofamil L，et al. Agriculture classification using POLSAR Data[C]. ESA Special Publication. ESA Special Publication，2005：32.
[12.8] Silva W B，Freitas C C，Sant'Anna S J S，et al. Classification of Segments in PolSAR Imagery by Minimum Stochastic Distances Between Wishart Distributions[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing，2013，6(3)：1263-1273.
[12.9] Lee J S，Grunes M R，Ainsworth T L，et al. Unsupervised classification using polarimetric decomposition and the complex Wishart classifier. IEEE Trans Geosci Remote Sens[C]. Geoscience and Remote Sensing Symposium Proceedings，1998. IGARSS '98. 1998 IEEE International. 1998：2178-2180 vol. 4.
[12.10] Fang C，Wen H. A new classification method based on Cloude Pottier eigenvalue/eigenvector decomposition[C]. IEEE International Geoscience and Remote Sensing Symposium. IEEE，2005：4 pp.
[12.11] Fang S，Hirose A. Use of Poincare sphere parameters for fast supervised PolSAR land classification[C]. IEEE International Geoscience and Remote Sensing Symposium. IEEE，2013：3175-3178.
[12.12] Yamaguchi Y，Ishido M，Yamada H，et al. A four-component decomposition of POLSAR image [C]. Geoscience and Remote Sensing Symposium，2005. IGARSS '05. Proceedings. 2005 IEEE International. IEEE，2005：4073-4076.
[12.13] Chen B，Wang S，Jiao L，et al. A Three-Component Fisher-Based Feature Weighting Method for Supervised PolSAR Image Classification[J]. IEEE Geoscience & Remote Sensing Letters，2015，12(4)：731-735.
[12.14] Xie W，Jiao L，Zhao J. PolSAR Image Classification via D-KSVD and NSCT-Domain Features Extraction[J]. IEEE Geoscience & Remote Sensing Letters，2016，13(2)：1-5.
[12.15] Jiao L，Liu F. Wishart Deep Stacking Network for Fast POLSAR Image Classification[J]. IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society，2016，25(7)：1-1.
[12.16] De S，Bhattacharya A. Urban classification using PolSAR data and deep learning[C]. IEEE Geoscience and Remote Sensing Society，the International Geoscience and Remote Sensing Symposium. IEEE，2015：353-356.
[12.17] Cheng J，Ji Y，Liu H. Segmentation-Based PolSAR Image Classification Using Visual Features：RHLBP and Color Features[J]. Remote Sensing，2015，7(5)：6079-6106.
[12.18] Zhang L，Ma W，Zhang D. Stacked Sparse Autoencoder in PolSAR Data Classification Using Local Spatial Information[J]. IEEE Geoscience & Remote Sensing Letters，2016，13：1-5.
[12.19] Gomez L，Alvarez L，Mazorra L，et al. Classification of PolSAR imagery by solving a diffusion-reaction system[C]. International Work Conference on Bioinspired Intelligence. IEEE，2015：73-80.
[12.20] Song H，Yang W，Xu X，et al. Unsupervised PolSAR Imagery Classification Based On Jensen-Bregman LogDet Divergence[C]. 2014 European Conference on Synthetic Aperture Radar；Proceedings of VDE，2014：1-4.
[12.21] 巫兆聪，欧阳群东，胡忠文. 应用分水岭变换与支持向量机的极化 SAR 图像分类[J]. 武汉大学学报信息科学版，2012，37(1)：7-10.
[12.22] 汪洋，鲁加国，张长耀. 基于支持向量机的极化 SAR 图像分类[J]. 无线电工程，2007，37(4)：

11-13.

[12.23] 滑文强，王爽，侯彪. 基于半监督学习的 SVM-Wishart 极化 SAR 图像分类方法[J]. 雷达学报，2015(1)：93-98.

[12.24] 陈博，王爽，焦李成，等. 贝叶斯集成框架下的极化 SAR 图像分类[J]. 西安电子科技大学学报(自然科学版)，2015，42(2)：45-51.

[12.25] Zou B，Li H，Zhang L. POLSAR image classification using BP neural network based on Quantum Clonal Evolutionary Algorithm[J]. 2010，38(1)：1573-1576.

[12.26] Wei B，Yu J，Wang C，et al. PolSAR image classification using a semi-supervised classifier based on hypergraph learning[J]. Remote Sensing Letters，2014，5(4)：386-395.

[12.27] Kiranyaz S，Ince T，Uhlmann S，et al. Collective Network of Binary Classifier Framework for Polarimetric SAR Image Classification：An Evolutionary Approach[J]. IEEE Transactions on Systems Man & Cybernetics Part B Cybernetics A Publication of the IEEE Systems Man & Cybernetics Society，2012，42(4)：1169-1186.

[12.28] Wang Y，He C，Tu F，et al. PolSAR image classification using feature fusion algorithm based on feature selection and bilayer SVM[J]. Journal of Wuhan University，2015，40(9)：1157-1162.

[12.29] Bai Y，Yang W，Xia G S，et al. A novel polarimetric-texture-structure descriptor for high-resolution PolSAR image classification[C]. IEEE International Geoscience and Remote Sensing Symposium. IEEE，2015：1136-1139.

[12.30] Li P X，Sun W D，Yang J，et al. High Resolution PolSAR Image Classification Based on Genetic Algorithm and Support Vector Machine [J]. ISPRS-International Archives of the Photogrammetry，Remote Sensing and Spatial Information Sciences，2013，40(7)：67-71.

[12.31] Krylov V A，Moser G，Serpico S B，et al. Enhanced Dictionary-Based SAR Amplitude Distribution Estimation and Its Validation With Very High-Resolution Data[J]. IEEE Geoscience & Remote Sensing Letters，2011，8(1)：148-152.

[12.32] Cintra R J，Rêgo L C，Cordeiro G M，et al. Beta generalized normal distribution with an application for SAR image processing[J]. Statistics，2014，48(2)：279-294.

[12.33] Liu M，Wu Y，Zhang P，et al. SAR Target Configuration Recognition Using Locality Preserving Property and Gaussian Mixture Distribution[J]. Geoscience & Remote Sensing Letters IEEE，2013，10(2)：268-272.

[12.34] Peng Y，Chen J，Xu X，et al. SAR Images Statistical Modeling and Classification Based on the Mixture of Alpha-Stable Distributions[J]. Remote Sensing，2013，5(5)：2145-2163.

[12.35] Zhang H，Wu Q M J，Nguyen T M，et al. Synthetic Aperture Radar Image Segmentation by Modified Student's t-Mixture Model[J]. Geoscience & Remote Sensing IEEE Transactions on，2014，52(7)：4391-4403.

[12.36] Zhou X，Peng R，Wang C. A Two-Component K-Lognormal Mixture Model and Its Parameter Estimation Method[J]. IEEE Transactions on Geoscience & Remote Sensing，2015，53(5)：2640-2651.

[12.37] Li H C，Krylov V A，Fan P Z，et al. Unsupervised Learning of Generalized Gamma Mixture Model With Application in Statistical Modeling of High-Resolution SAR Images[J]. IEEE Transactions on Geoscience & Remote Sensing，2016，54(4)：2153-2170.

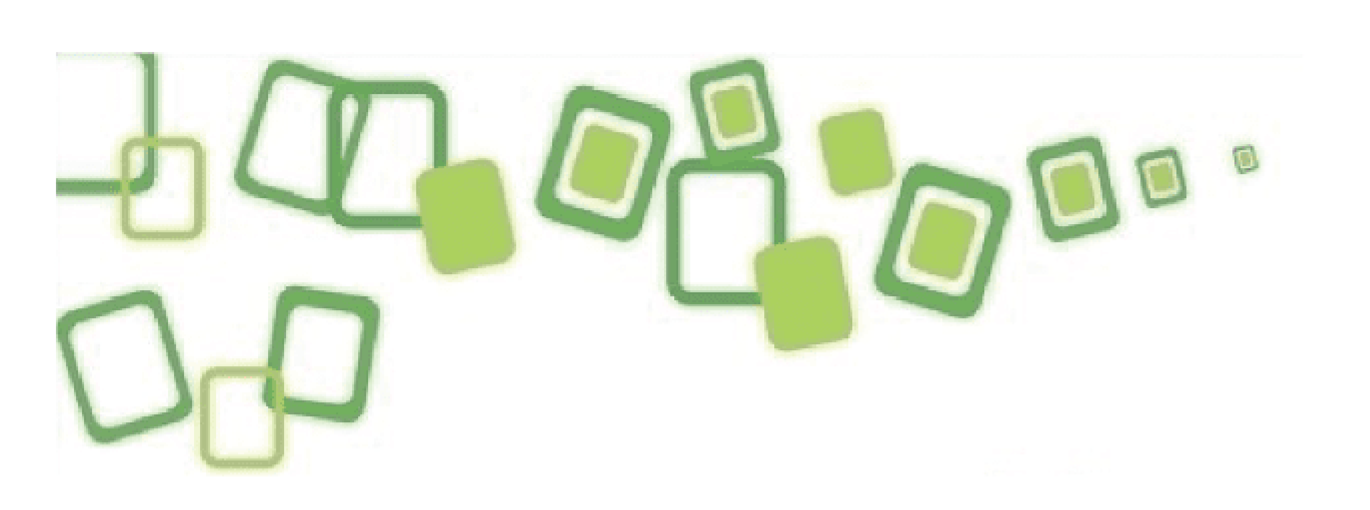
[12.38] Yang W，Dai D，Triggs B，et al. SAR-based terrain classification using weakly supervised hierarchical Markov aspect models. [J]. IEEE Transactions on Image Processing，2012，21(9)：4232.

[12.39] Kayabol K，Zerubia J. Unsupervised amplitude and texture classification of SAR images with multinomial latent model[J]. IEEE Transactions on Image Processing，2013，22(2)：561-572.

[12.40] He C，Zhuo T，Ou D，et al. Nonlinear Compressed Sensing-Based LDA Topic Model for Polarimetric SAR Image Classification[J]. IEEE Journal of Selected Topics in Applied Earth

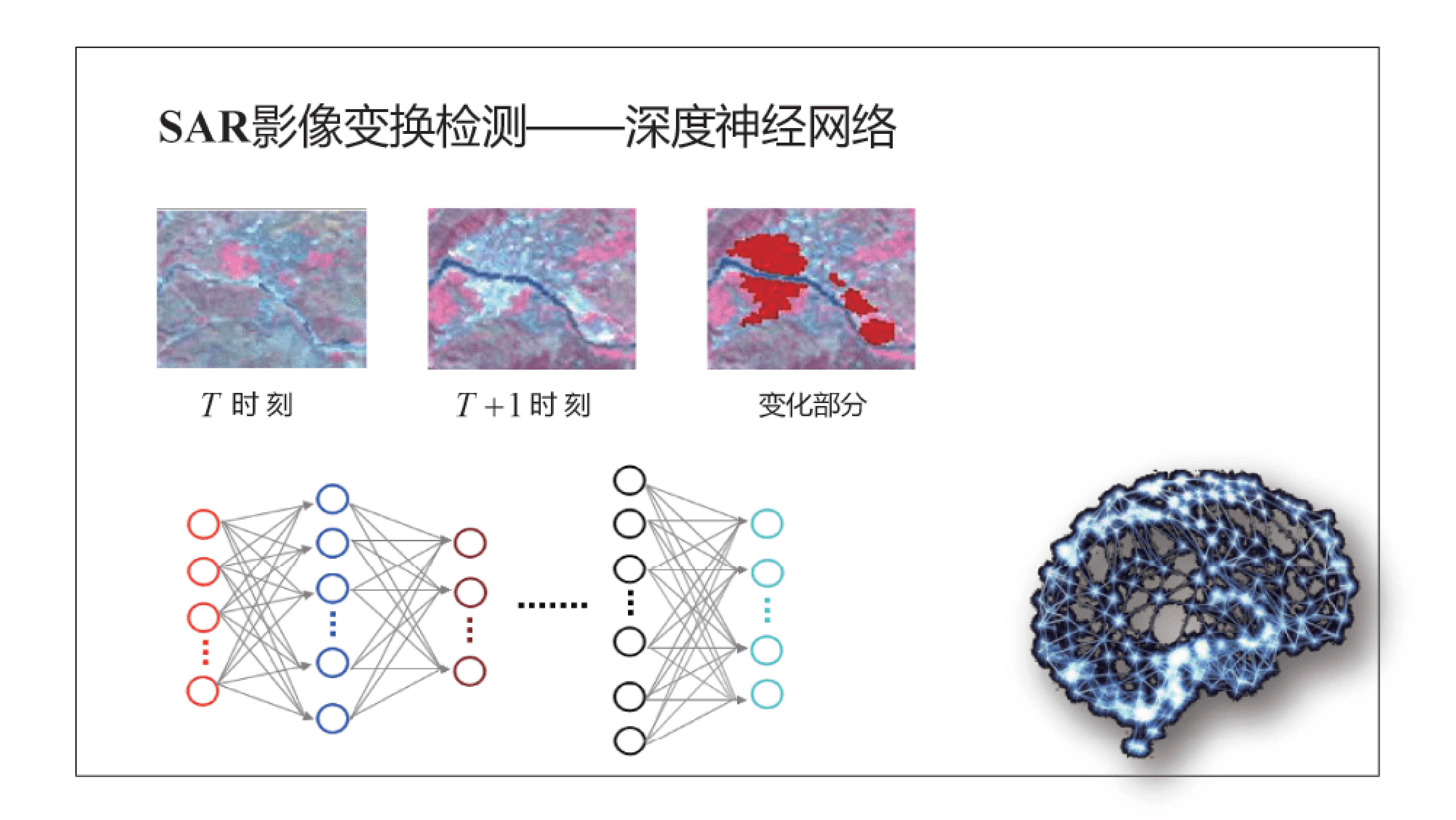
Observations & Remote Sensing, 2014, 7(3): 972-982.
[12.41] Haralick R M, Shanmugam K, Dinstein I. Textural Features for Image Classification[J]. Systems Man & Cybernetics IEEE Transactions on, 1973, 3(6): 610-621.
[12.42] Soh L K, Tsatsoulis C. Texture analysis of SAR sea ice imagery using gray level co-occurrence matrices[J]. IEEE Transactions on Geoscience & Remote Sensing, 1999, 37(2): 780-795.
[12.43] Champion I, Germain C, Costa J P D, et al. Retrieval of Forest Stand Age From SAR Image Texture for Varying Distance and Orientation Values of the Gray Level Co-Occurrence Matrix [J]. IEEE Geoscience & Remote Sensing Letters, 2014, 11(1): 5-9.
[12.44] Ressel R, Frost A, Lehner S. A Neural Network-Based Classification for Sea Ice Types on X-Band SAR Images[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing, 2015, 8(7): 3672-3680.
[12.45] 薄华，马缚龙，焦李成. 图像纹理的灰度共生矩阵计算问题的分析[J]. 电子学报，2006，34(1)：155-158.
[12.46] 刘丽，匡纲要，等. 图像纹理特征提取方法综述[J]. 中国图象图形学报，2009，14(4)：622-635.
[12.47] Li J, Allinson N M. A comprehensive review of current local features for computer vision[J]. Neurocomputing, 2008, 71(10-12): 1771-1787.
[12.48] Clausi D A, Deng H. Design-based texture feature fusion using Gabor filters and co-occurrence probabilities[J]. IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society, 2005, 14(7): 925-36.
[12.49] Kandaswamy U, Adjeroh D A, Lee M C. Efficient Texture Analysis of SAR Imagery[J]. IEEE Transactions on Geoscience & Remote Sensing, 2015, 43(9): 2075-2083.
[12.50] Dumitru C O, Datcu M. Information Content of Very High Resolution SAR Images: Study of Feature Extraction and Imaging Parameters[J]. IEEE Transactions on Geoscience & Remote Sensing, 2013, 51(8): 4591-4610.
[12.51] 焦李成，冯婕，刘芳，杨淑媛，等. 高分辨遥感影像学习与感知[M]. 北京：科学出版社，2017.
[12.52] 焦李成，张向荣，侯彪，王爽，刘芳，等. 智能 SAR 图像处理与解译[M]. 北京：科学出版社，2008.
[12.53] 隋艳立. 基于深度支持向量机的极化 SAR 图像分类[D]. 西安电子科技大学，2014.
[12.54] 刘宸荣. 基于自编码神经网络的极化 SAR 影像地物分类[D]. 西安电子科技大学，2014.
[12.55] 韩佳敏. 基于深度 RBF 网络的 SAR 影像地物分类[D]. 西安电子科技大学，2014.
[12.56] 刘贺. 基于深层特征学习和稀疏表示的 SAR 图像分类[D]. 西安电子科技大学，2015.
[12.57] 龙贺兆. 基于稀疏深层网络的 SAR 图像分类方法[D]. 西安电子科技大学，2015.
[12.58] 许声红. 基于自适应自编码和超像素的 SAR 图像分类[D]. 西安电子科技大学，2015.
[12.59] 汤玫. 基于稀疏编码字典和深度学习的极化 SAR 影像地物分类[D]. 西安电子科技大学，2015.
[12.60] 普亚如. 基于 CNN 特征学习和 SVM 的极化 SAR 影像地物分类[D]. 西安电子科技大学，2015.
[12.61] 张曼. 基于增量学习和深度稀疏滤波的复杂场景多类标分类[D]. 西安电子科技大学，2015.
[12.62] 鄢蕾. 基于深度 ICA 网络的极化 SAR 影像地物分类方法研究[D]. 西安电子科技大学，2015.
[12.63] 熊莎琴. 基于稀疏编码和 SVM 的极化 SAR 影像地物分类方法研究[D]. 西安电子科技大学，2015.
[12.64] 张丹. 基于 SIFT 特征表示和稀疏编码的多标记场景分类[D]. 西安电子科技大学，2015.
[12.65] 高蓉. 面向极化 SAR 地物分类的稀疏深度网络[D]. 西安电子科技大学，2015.
[12.66] 王昭. 基于目标分解与机器学习的极化 SAR 图像地物分类[D]. 西安电子科技大学，2015.
[12.67] 张亚楠. 基于深度脊波神经网络的极化 SAR 影像地物分类[D]. 西安电子科技大学，2015.
[12.68] 白雪莹. 基于深度 RPCA 网络的极化 SAR 影像地物分类[D]. 西安电子科技大学，2015.
[12.69] 牟洋. 基于稀疏表示分类器的极化 SAR 图像地物分类[D]. 西安电子科技大学，2015.
[12.70] 闫晓莉. 基于素描稀疏表示和低秩分解的 SAR 图像目标检测[D]. 西安电子科技大学，2016.
[12.71] 马丽媛. 基于深度 Contourlet 卷积网络的遥感图像地物分类[D]. 西安电子科技大学，2017.

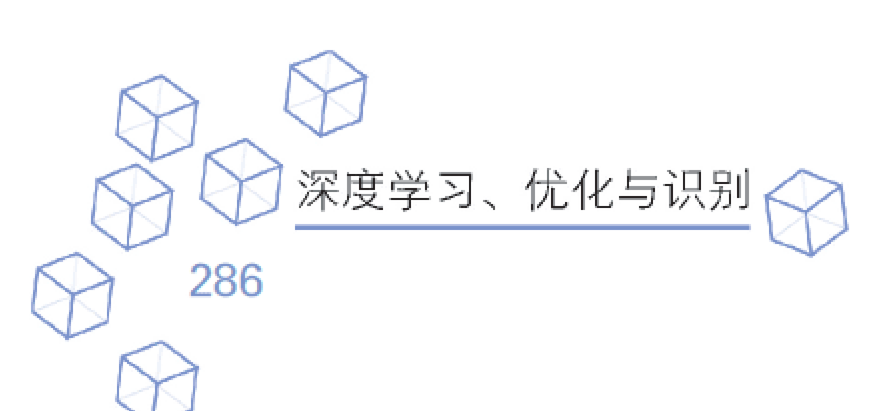
第13章



基于深度神经网络的SAR影像的变化检测

CHAPTER 13





13.1 数据集特点及研究目的

合成孔径雷达(SAR)影像变化检测是指通过对不同时期同一区域的 SAR 影像进行比较分析,根据影像之间的差异得到我们所需要的地物或目标的变化信息。现代遥感技术的飞速发展为变化检测提供了一种便捷的途径,遥感数据成为变化检测的主要数据源。与可见光和红外遥感相比,微波遥感具有无可比拟的优点:微波能穿透云雾、雨雪,具有全天候、全天时的工作能力;微波对地物有一定穿透能力;采用侧视方式成像,覆盖面积大。正是这些优点,使得 SAR 图像日益成为变化检测的重要数据源。

SAR 变化检测技术的需求日益广泛。目前,全球环境变化加剧,城市急速发展,洪水、地震等自然灾害时有发生,这些都需要及时掌握相关动态信息,为相关决策部门提供支持,而 SAR 的种种优点为快速响应提供了技术支持和应急保障。

13.1.1 研究目的

如图 13.1 为 SAR 影像变化检测的一般流程图。主要分为 SAR 数据源获取、影像预处理、变化检测、精度估计四个步骤,下面简要介绍一下各步骤。

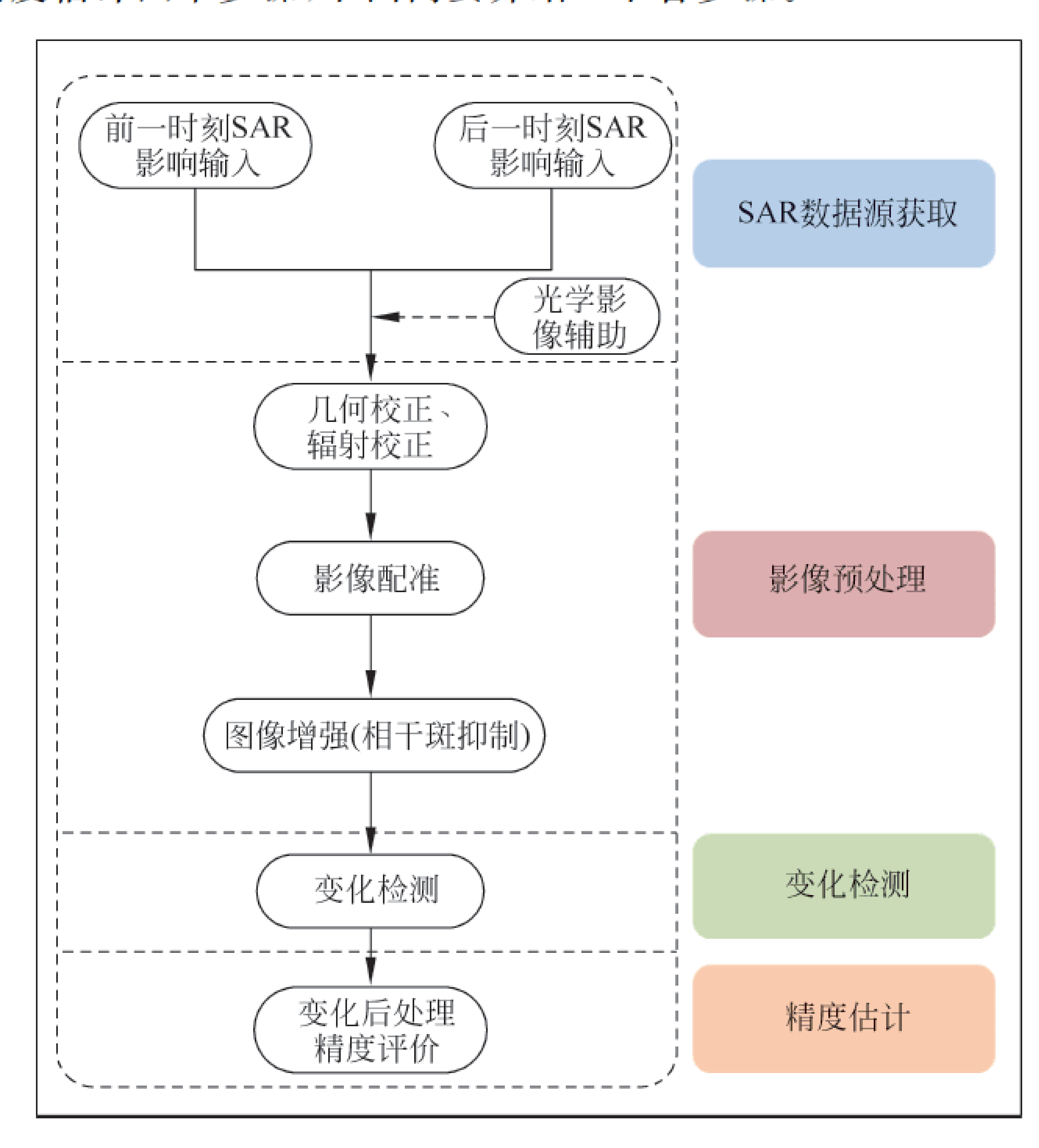


图 13.1 变化检测流程图

SAR数据源获取：对数据进行初步分析，获取图像的相关信息：SAR系统参数（分辨力、波段、波长、极化方式、像素间距）；数据场景的位置和方向；数据类型是原始数据还是图像数据，如果数据类型是原始数据，则需进行成像处理；数据的量化方式和精度；数据存储格式；数据获取的时间，然后从存储介质中获取数据。

影像预处理：由于传感器差异和其他干扰的影响，在对SAR影响进行数据分析之前，通常需要进行一定的预处理修正，典型的预处理修正包括降噪、辐射校正、传感器校准、地形校正、几何校正和图像配准等。

几何校正：由于SAR图像在获取过程中会受到传感器位置和运动状态变化、地形起伏、地球表面曲率、大气折射和地球自转等因素的影响，产生几何形变，因此需要对图像进行几何校正。几何校正的大致过程为：

（1）由图像中所含的几何畸变性质及用于校正的数据来确定校正的方法；

（2）确定校正公式和结构，例如图像坐标和地图坐标的变化式等，根据控制点数据求出校正的参数；

（3）验证校正方法、校正的有效性；

（4）重采样、内插。

辐射校正：在信号获取过程中，不同的传感器以及不同的地物特征等情况会对辐射值造成影响，为了客观评价地表的发射特征及辐射特征，必须对SAR图像进行辐射标准化，以使得两图未变化部分的灰度值大致相同。完整的辐射校正包括传感器校正、大气校正和地形校正等。辐射校正分为绝对辐射定标和相对辐射定标。绝对辐射定标就是使用卫星星历数据，反演内插SAR图像地物雷达后向散射截面值；而相对辐射定标，就是以一幅图像为基准，把其他数据序列集图像映射投影变换到基准亮度空间。一般的SAR数据文件中，都会给出校正参数值。

图像配准：根据主图像的几何特性，对两幅图像进行图像配准，使得主图像和辅图像的空间位置对应。

图像增强：对SAR图像进行增强处理可以突出有用信息，对相干斑噪声进行抑制以改善图像质量。图像增强的方法主要有对比度扩展、空间滤波、图像运算。其中空间滤波以重点突出图像上的某些特征为目的，如突出边缘或纹理等，主要包括平滑和锐化。常用的平滑运算包括均值平滑和中值平滑。锐化方法常用的有Roberts梯度、Sobel梯度、拉普拉斯算法和定向检测。锐化可以对图像进行边缘增强和边缘提取。对SAR图像而言，主要是进行相干斑噪声抑制。

最后，在校正的基础上，还需要通过对图像内容、特征、结构、关系、纹理及灰度等的对应关系、相似性和一致性进行分析，最终得到精确配准的SAR图像。

变化检测：变化检测是一个确定和评价各种地表现象随时间发生变化的过程。检测出两幅SAR图之间发生变化的区域。根据具体的应用，对检测结果进行分析。

精度估计：在SAR图像变化检测中，为了评估得到的变化检测图像的质量记忆算法的好坏，一般用检测精确率、虚警率、Kappa系数等指标进行评价。检测准确率是指把检测出



来的变化信息与地面真实变化进行比较，检测结果正确的百分比。虚警率是指实际未发生变化，却被检测为变化的像素个数所占的百分比。Kappa 系数用来估计变化检测得到的结果与地面真实变化二者之间的相似性程度，对于地面的变化情况，Kappa 系数比其他评价标准更敏感，更能反映检测性能的差异。Kappa 系数越大，则表明两幅图像的相似程度就越高，即变化检测得到的结果与地面真实变化越接近。Kappa 系数为 1，则说明完全一致。

变化信息的获取是变化检测过程中的核心和关键，目前所出现的各种变化检测方法也都是为了解决如何有效地从多时相图像中提取出地物的变化信息。虽然已经出现了各种各样的变化检测方法，但是它们基本上都是为了解决某个或某类问题而提出的，因此缺乏统一的描述。从不同的角度出发，可以进行不同的分类。如果从检测层次的角度出发，可以分为像素级变化检测、特征级变化检测和目标级变化检测；如果从应用的角度出发，可以分为基于土地覆盖的变化检测、基于人工地物的变化检测、基于土壤植被索引的变化检测等；如果从算法的角度出发，则可以分为基于图像代数运算、基于图像变换、基于图像分类以及基于图像结构特征分析的变化检测等。

在过去的几十年里，人们提出了很多 SAR 图像的变化检测方法。其中大部分是基于差异图的分析方法。这类方法通常包括三个主要步骤：图像的预处理、差异图的生成和分析。差异图分析是非常关键的步骤，它可以被看作是一个自动的分割过程——将差异图分成变化类和非变化类。最经典的分析差异图的方法包括：阈值法、聚类法等。阈值法通常自动地寻找一个固定常数，通常将差异图中的像素灰度值与该常数比较决定其变化类别。这种方法的主要缺点是其检测的正确性依赖于统计模型与实际数据分布的拟合度，并且很少考虑像素的邻域信息。而聚类法将差异图中的像素进行自动分组，使得同一类别像素之间的距离最近，而不同类别像素之间的距离最远。实现该过程需要建立目标方程，对聚类中心和隶属度进行迭代。而如何建立无偏袒的目标方程是这类方程所面临的瓶颈问题。

随着数据收购渠道和应用范围的不断增加，传统的方法不能满足更高的精度和更灵活的应用。SAR 图像中斑点乘性噪声的存在使得 SAR 图像很难被解译，这将干扰区域本身的变化情况，从而影响变化预测的判断。

深度学习被看作是神经网络的复兴，在近些年引起了人们的广泛关注。随着近年来深度学习的火热，深度学习成为一个新的机器学习方法并逐渐应用于视觉的各个领域。无论在语音识别、目标识别、多任务，还是在迁移学习等领域都取得了突破性进展。深度学习从数据出发，试图利用未知结构去发现良好的且具有较高层次的学习特征用以定义和表示较底层的特征。它包括一连串的处理单元，其目的是对未加工的输入信号进行不同形式的表示。深层网络可以通过不断地学习，在网络的输出层获得抽象且不变的特征，发现非局部结构，最终使得输出表示更加简化和鲁棒。

本章试着从深度学习的模型出发，为解决 SAR 图像的变化检测的相关问题提供一些思路。

13.1.2 数据基本特性

SAR 图像是利用微波遥感技术得到的，与光学手段得到的图像不同，它反映的是目标的无线电波散射特性，而不是光学特性。雷达图像依据回波信号的强弱形成，强弱程度决定了图像的灰度。某地区的回波信号强，反映在图像上，其对应位置的灰度就高；回波信号弱，灰度就低。SAR 图像与光学图像有以下区别：

- 微波具有穿透性，可以透过云层、地表观测，不受光线强度影响的特点，因此成像雷达具有全天候、全天时的工作能力，这是光学成像不具备的优点。而且成像雷达与光学传感器相比，具有更大的侦察范围，可以发现不容易被光学传感器发现的目标，得到大范围、高分辨率的图像。
- 由于成像雷达的回波信号需要进行一系列的复杂处理，因此与光学成像相比，设备更复杂、运算量更大。
- 成像雷达是相干处理系统，图像存在相干斑，这是雷达系统固有的缺点。相干斑导致图像质量下降，因此成像处理后，往往都需要进行相干斑抑制。光学成像由于原理不同，不存在这个问题。
- 雷达成像就是从回波信号中提取目标的后向散射系数，所以图像反映的是被测地域的电磁特性，而光学成像依据的是普通的反射。两部分区域光学特性不同，但后向散射系数可能相同，因此雷达图像不能区分这两个区域，在光学图像上区别却很明显。
- 光学图像通常是垂直照射地面所得，成像雷达一般则是侧视成像。雷达波束以一定的俯角照射被测绘的地域，使得雷达图像具有阴影、迎坡缩短等固有特征。与光学图像相比，雷达图像的轮廓比较清楚，有较好的对比度。
- 在知道雷达的各种参数（如高度、入射角）后，对雷达图像进行插值等处理可以得到相同的比例尺表示，图像不会发生畸变。光学图像由于光在成像透镜的光轴周围的折射率不同，使得图像出现畸变，如远离轨迹处的图像被压缩。

SAR 是一种主动式微波成像传感器，SAR 主要利用脉冲压缩技术和合成孔径原理，使得距离分辨率和方位分辨率分别加以提高，进而获取大面积的高分辨率的遥感影像。SAR 图像由于其全天时、全天候和一定穿透性的特殊成像特性，为地球空间信息的获取提供了一个有效途径，其在目标跟踪、自然灾害检测以及农作物生长监测等众多应用中都发挥了重要作用。近年来 SAR 系统的飞速发展，使得 SAR 数据的获取更为便捷。特殊的成像特性使得其相对于光学图像可以在极端天气情况下发挥重要作用。随着获取时间的缩短，同一地区不同时相 SAR 数据为变化检测技术的开展提供了数据支持，SAR 变化检测技术逐渐成为遥感应用研究的热点。然而 SAR 成像系统基本分辨单元内地物的随机后向散射，使得相位角失去了连续性，在影像上表现为颗粒状信号相关的强度畸变，即产生相干斑乘性噪声。变化与非变化类的相关统计项很难得到准确估计，在相干斑抑制和细节保持上存在矛盾，故进行精确的多时相 SAR 影像的变化检测存在较大的困难。以下从成像机理得出影响 SAR

图像变化检测的图像特点：几何特性、统计分布特性、相干斑噪声特性。

1. SAR 图像的几何特性

一般来说，SAR 图像是地面的斜距投影。在斜距投影方式的 SAR 图像上，距离向的比例尺是变化的，其随着侧视角的增大而变大，同样大小的地面目标，距离天线正下方越近，在图像上的尺寸越小，即图像上近地点被压缩，远地点被拉长。方位向上的比例尺是固定的，它取决于平台的飞行参数。在知道雷达的各种参数后，对雷达图像进行插值等处理可以在距离向、方位向上得到相同的比例尺表示。

当侧视角大于地面坡度，雷达波束照射到位于天线同一侧的斜面时，波束到达顶部的斜距和到达底部的斜距差要小于实际地面距离，造成在图像上的斜面长度被缩短了，即所谓的“透视收缩”；同样对于背向天线的地面斜坡，也会出现透视收缩，不过斜坡看起来被拉长了。当侧视角小于地面坡度，雷达波束到达斜坡顶部的时间比到达底部的时间短时，和中心投影时的点位关系相比，会出现顶部图像和底部图像颠倒的现象，称为“叠掩倒像”。由于地形一般是非线性变化，图像会产生偏扭、弯曲等形变。另外，在实际应用中，卫星、飞机的飞行轨迹并不能保证是直线。在方位向，图像也会有偏扭、弯曲等形变，因此对 SAR 图像进行几何校正时，需要采用多项式几何校正。

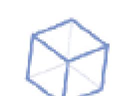
2. SAR 图像的统计分布特性

SAR 图像的分布特性无论对 SAR 图像相干斑抑制问题，还是 SAR 图像中变化检测的问题，都很重要。研究表明，在宽带高分辨雷达体制下，场景杂波特性可以从幅度统计特性和相关特性(功率谱特性)两方面同时进行描述，即把场景杂波描述为具有某种幅度分布的相关随机过程。研究人员提出了几种分布来描述 SAR 图像的统计特性。对于低分辨 SAR 图像的均匀区域，幅度服从 Rayleigh 分布，强度服从负指数分布。对于分辨率较高的 SAR 图像，广泛应用的是对数正态分布和 Weibull 分布。近年来，国内外研究工作者提出了用 K 分布、Gamma 分布来描述 SAR 图像的统计特性。目前，研究人员把宽带杂波中的相干斑成分和纹理分布都看作广义 Gamma 分布，这样就得到基于乘积模型的广义复合分布模型，常用的指数、高斯、瑞利、对数正态、Weibull、Gamma 分布都是广义复合分布的特例。

3. SAR 图像的相干斑噪声特性

在 SAR 图像中，对一个同质区域，即它的后向散射的物理特征是平稳的，看上去却显得好像是由许多不同强度的像素组成的，这些像素点的亮度值远不是大致相同，而是极其发散，对于那些视数很小的图像尤其如此。对于单视图像，后向散射强度为零的概率最大。另外，像素的方差随着雷达平均反射系数的增加而增加。这种发散性使 SAR 图像看起来是一种噪声极强的颗粒状斑点，这种现象来源于相干斑(speckle)，也叫斑点噪声。

SAR 发射的是相干电磁波，雷达脉冲照射的地表单元都包含了很多的散射点，这一单元的总的回波是各个散射点的相干叠加，而每个散射点回波的相位同传感器与该点的距离




有关,当传感器移动时,所有单元内的散射点的回波相位都发生变化,幅度也随之变化,这样当传感器移动中连续观测同一地表单元时将得到不同的幅度,这种幅度的变化称为衰落,同样地,具有相同后向散射截面(RCS)的两个观测单元,如果细微特征有差异,它们的回波信号也会不同,这样本来具有常数后向散射截面的图像的同质区域,像素间会出现亮度变化。

对分布目标,可以认为 SAR 图像中的单个分辨单元有许多离散的散射源,当电波作用于目标时,每个散射源都产生一个后向散射波,这些散射波的相位和幅度与该散射源有关,因此该分辨单元的总回波是各个散射源回波之和。

$$z = A \cdot e^{j\cdot\phi} = \sum_{k=1}^{n} A_k \cdot e^{j\cdot\phi_k} \tag{13.1}$$

其中 A 是回波幅度,ϕ 是回波相位,A_k 和 ϕ_k 分别是第 k 个散射单元反射雷达电磁波的幅度和相位。从式(13.1)可以看出是观察的信号受不同散射源间的相位差异造成的干涉效应的影响。实际上,相干斑可以认为是一种干涉现象。观察数据的噪声特性的根本起因是相位项 ϕ_k。

SAR 图像固有的相干斑噪声严重影响了图像质量,不能正确反映地物目标的散射特性,给提取图像中目标的信息造成了很大的困难。因而对 SAR 图像应用前,都要进行相干斑抑制,以提高图像质量,这是 SAR 图像后处理的必不可少的环节,也是 SAR 图像处理的重要课题之一。美国于 1978 年最先发射了全球第一颗装在 SAR 传感器的人造地球卫星(SEASAT-A),获得了大量的地表信息,极大地推动了 SAR 理论与技术的发展。此后,越来越多的国家和地区对星载 SAR、机载 SAR 进行研制或发射。目前,国际上主要使用的 SAR 卫星系统及参数列于表 13.1 中。

表 13.1　国际上主要卫星系统以及相关参数

SAR 卫星	国家或机构	轨道高度/km	波长/mm	入射角	空间分辨率/m	幅宽/km	服役时间段
ERS-1/2	欧空局	790	56	23°	20×20	100	1992—2001
JERS-1	日本	570	235	35°	18×18	75	1992—1998
RADARSAT-1	加拿大	790	56	23°～56°	10～100	50～500	1995—2013
ENVISAT	欧空局	790	56	15°～45°	20～100	100～400	2002—2010
ALOS	日本	700	235	8°～60°	7～100	20～350	2006—2011
RADARSAT-2	加拿大	798	56	20°～58°	3～100	25～500	2007 年至今
TerraSAR-X	德国	514	31	20°～55°	1～16	10～100	2007 年至今
TanDEM-X	德国	514	31	20°～55°	1～16	10～100	2007 年至今
COSMO-SkyMed	意大利	620	31	20°～55°	1～20	10～200	2009 年至今
SENTINEL-1	欧空局	693	56	20°～45°	5～80	20～400	2014 年至今
ALOS-2	日本	628	238	14°	3～100	25～350	2014 年至今

我国对星载极化干涉雷达系统的研制还处在起步阶段,已经实施了环境与减灾卫星计划,于 2012 年底成功发射了搭载 S 波段单极化 SAR 系统的环境 1 号 C(HJ-1C)卫星。随着

研究的深入，SAR 系统开始由单波段、单极化、固定入射角、单工作模式逐渐向多波段、多极化、多入射角、多工作模式和多平台方向发展，使得 SAR 在遥感应用领域起到越来越重要的作用，与此同时应用于 SAR 的变化检测技术也获得了快速发展。

13.1.3 典型数据集

下面介绍几组常用的 SAR 图像数据集：

(1) Bern 地区数据集的原始影像分别在 1999 年 4 月和 1999 年 5 月通过欧洲遥感 2 号星载 SAR 传感器在瑞士 Bern 地区获得，在此时间段内，泛滥的 Aare 河洪水将 Thun 和 Bern 两座城市的部分地区淹没，Bern 机场则是彻底被洪水淹没，前一时刻的 SAR 影像显示了洪水尚未发生时的情形，后一时刻的 SAR 影像中可以清楚地看出当时泛滥的洪水，两幅影像的尺寸均为 301×301，如图 13.2(a)和图 13.2(b)所示；而变化参考图 13.2(c)通过结合当地真实的陆地信息和专家知识得到。

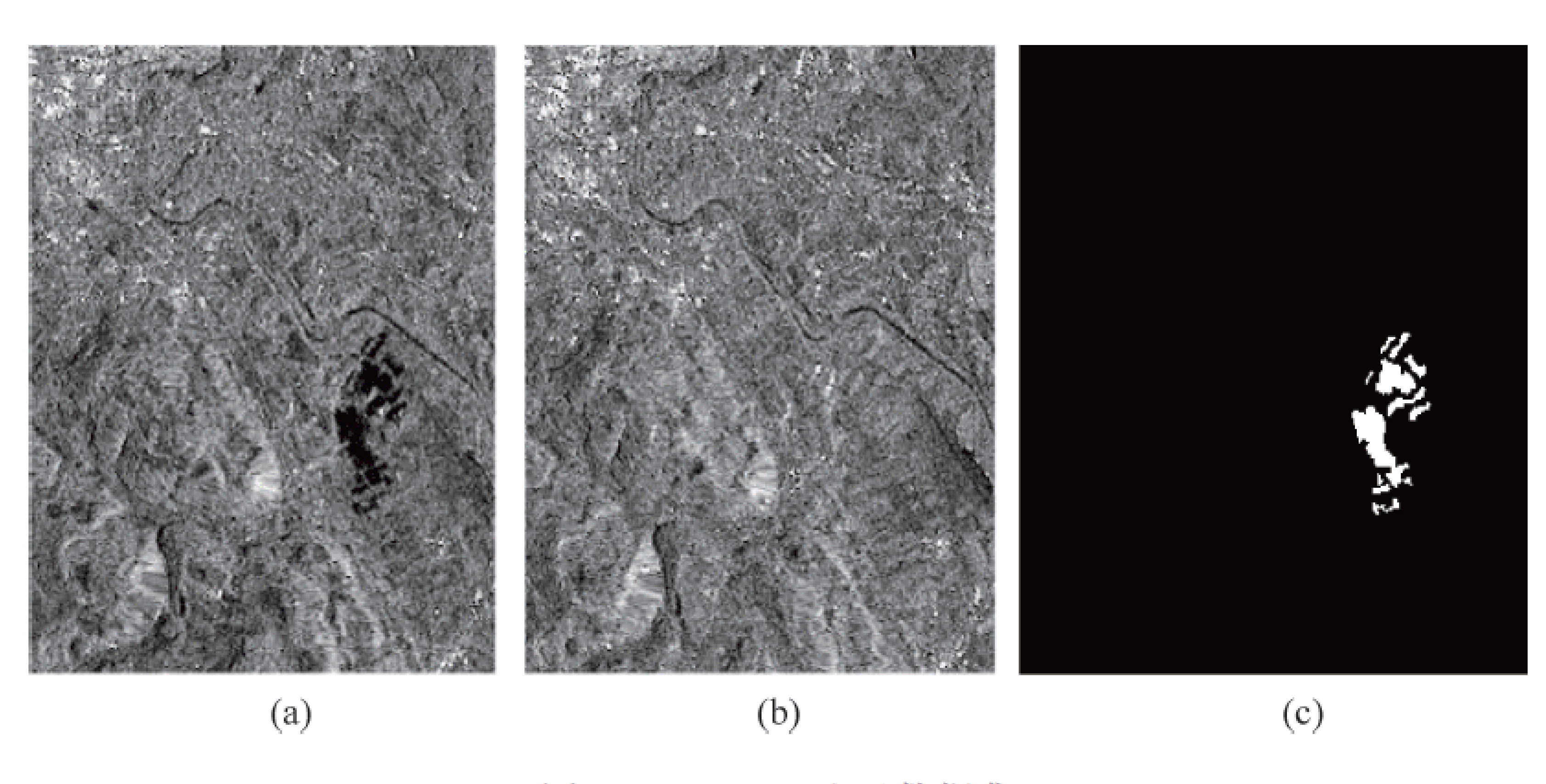


图 13.2　Bern 地区数据集

(2) Ottawa 数据集是由 RADARSAT 卫星分别在 1997 年 5 月和 1997 年 8 月拍摄，分辨率为 12m，影像大小为 290×350。该数据集反应的是加拿大 Ottawa 地区受雨季影像其地表变化情况，此时间段正值 1997 年的雨季过后，河道明显变窄。从图 13.3 中可以清楚地看到河水退去后露出的大范围陆地区域，变化参考图结合当地真实的陆地信息和专家知识得到。

(3) 黄河口数据集是由 Radarsat-2 卫星分别在 2008 年 6 月和 2009 年 6 月拍摄，分辨率为 8m，影像大小为 7666×7692，如图 13.4(a)和图 13.4(b)所示。其中，2008 年的影像是四视图，2009 年的影像是单视图，这也表明噪声对两幅图像的影响是不同的。图 13.5 和图 13.6 都是从黄河流域中选取的一部分，其中图 13.5(c)和图 13.6(c)为人工标记的参考图。

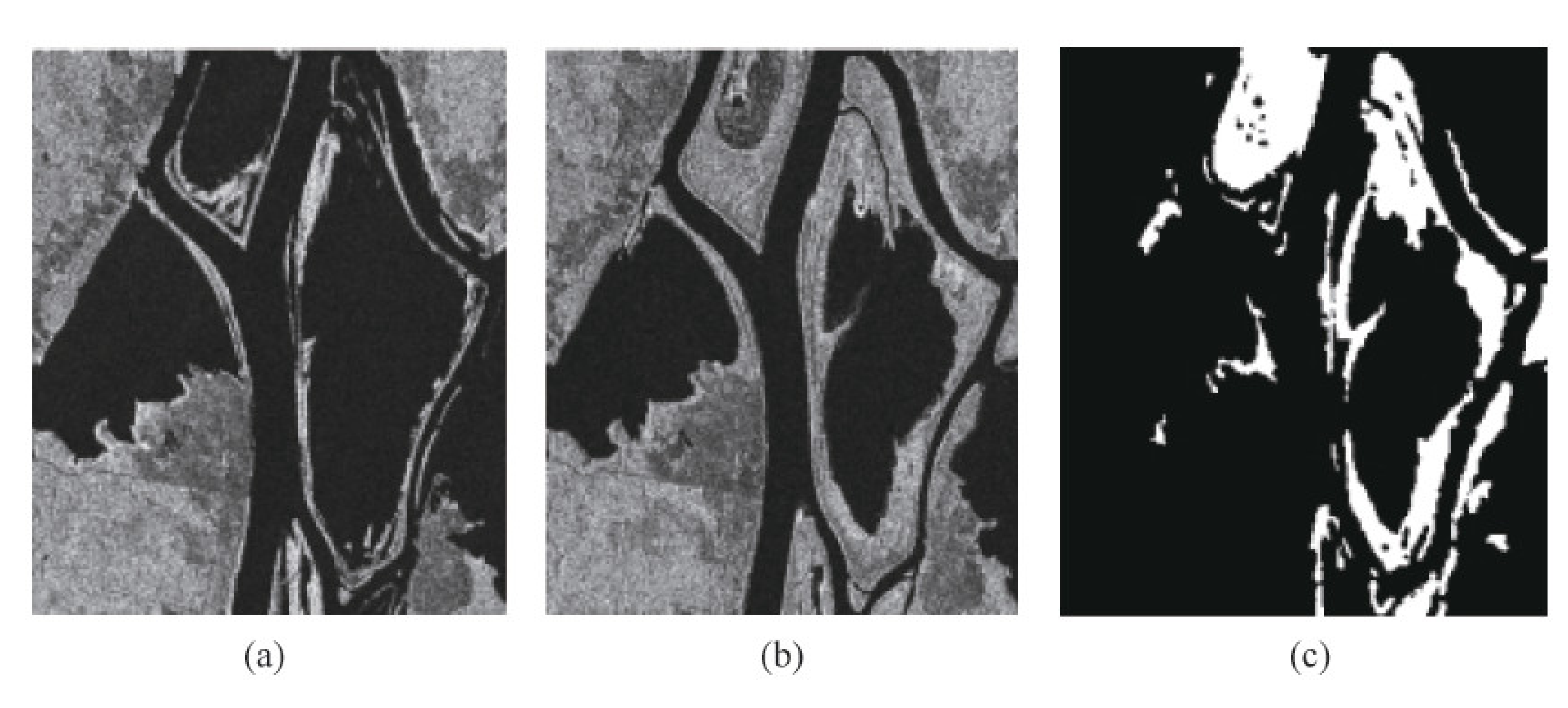


图 13.3　Ottawa 数据集

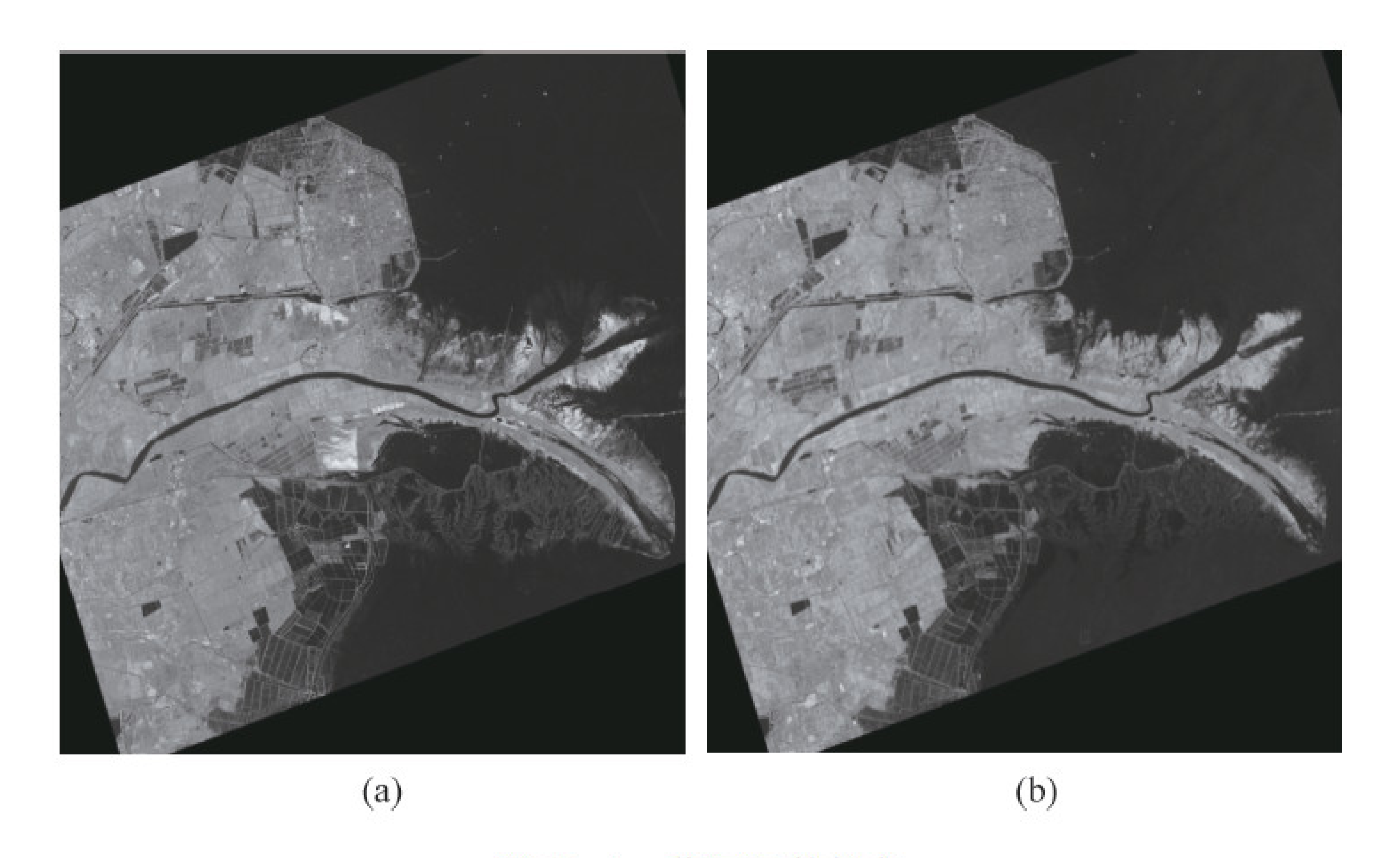


图 13.4　黄河口数据集

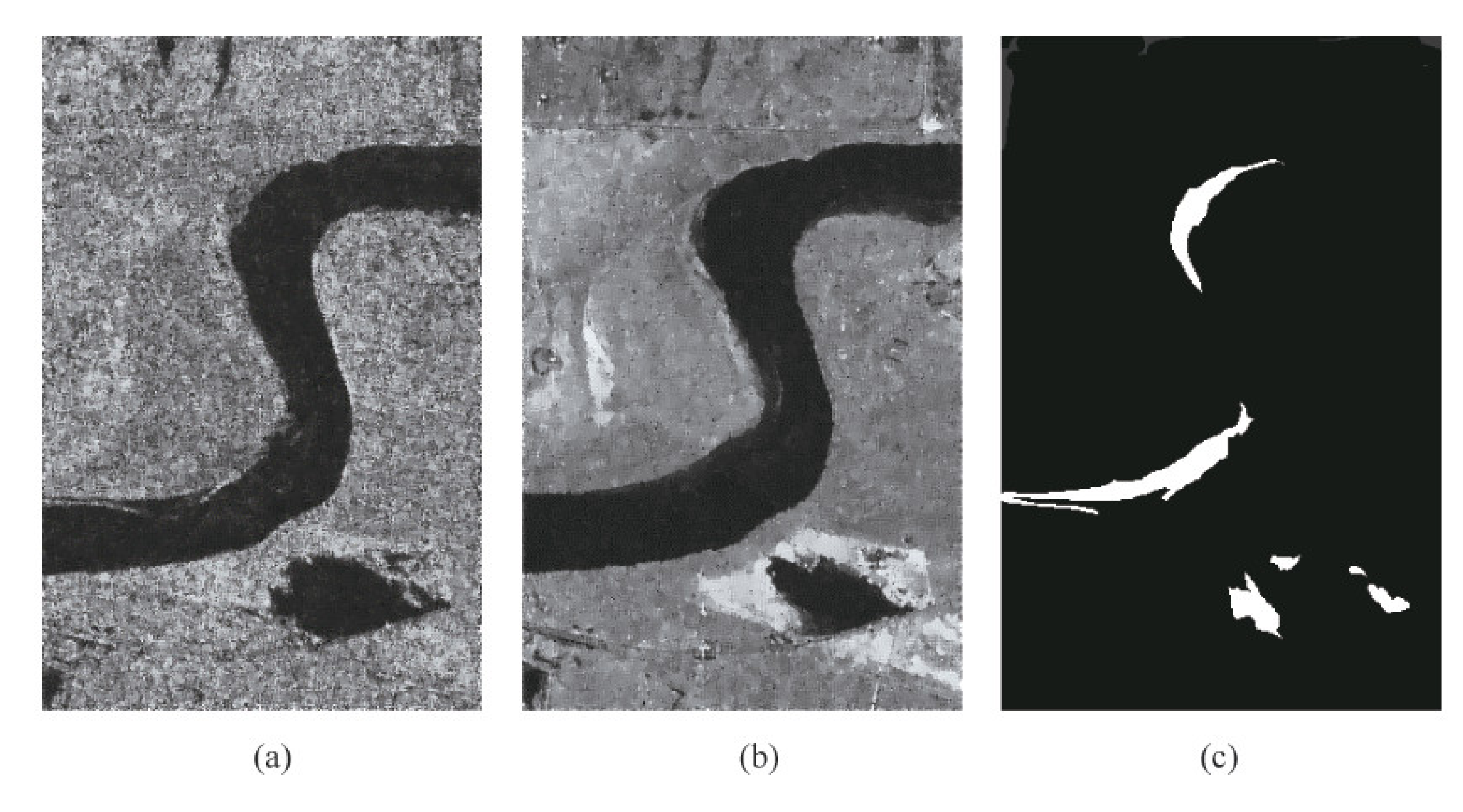


图 13.5　黄河口数据集中选取的一部分，记为黄河口 S 形

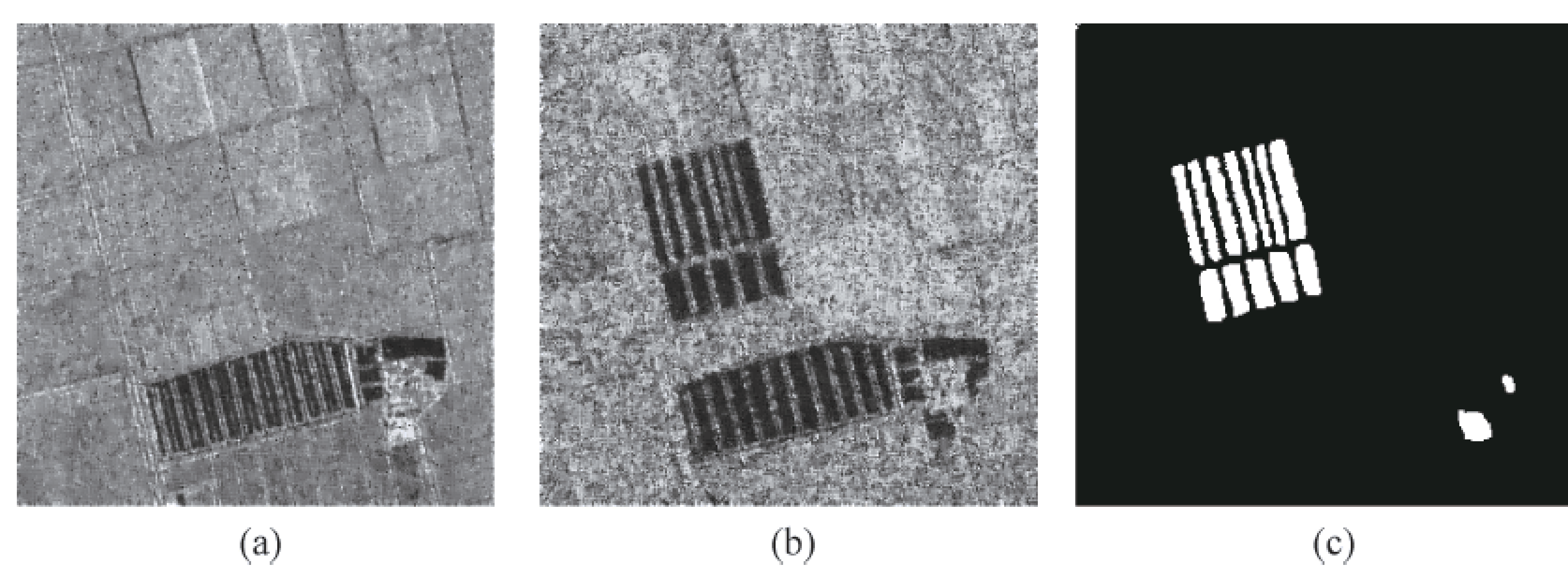


图 13.6　黄河口数据集中选取的一部分，记为黄河口农田

13.2　基于深度学习和 SIFT 特征的 SAR 图像变化检测

本节介绍一种基于深度学习和 Scale-invariant feature transform（SIFT）特征的 SAR 图像变化检测方法，以实现对 SAR 图像变化区域的准确检测。该方法结合了深度学习与 SIFT 特征两种方法，直接通过 SIFT 特征训练深度神经网络，由于 SIFT 特征可以反映图像的局部特征，对图像旋转、尺度缩放以及亮度变化均具有不变性，而且对仿射变换和噪声也保持一定程度的稳定性，因而可以作为深度神经网络的可靠训练样本。该方法思路简单明确，通过有效利用原始图像的特征提高了变化检测的精度。

实现上述目的的思路是：首先运用尺度不变特征变换方法提取原图像的 SIFT 特征，将此作为训练样本，训练一个深度神经网络。再利用对数比值法得到原始图像的差异图，提取该差异图每个像素点的邻域特征，以此作为测试数据，输入到训练好的深度神经网络里进行测试，输出最终的变化检测结果。

这里简要介绍 SIFT 算法：SIFT 是 David Lowe 于 1999 年提出的局部特征描述算子，并于 2004 年进行了更深入的发展和完善。Mikolajczyk 对包括 SIFT 算子在内的十种局部描述算子所做的不变性对比实验中，SIFT 及其扩展算法已被证实在同类描述算子中具有最强的鲁棒性。

SIFT 特征的生成一般包括以下四个步骤：

(1) 高斯差分(DOG)尺度空间的生成；

(2) 关键点的精确定位；

(3) 特征主方向的分配；

(4) 特征描述符的生成。

总体来说，SIFT 算子具有以下特点：

- SIFT 特征是图像的局部特征，对平移、旋转、尺度缩放、亮度变化、遮挡和噪声等具有良好的不变性，对视觉变化、仿射变换也保持一定程度的稳定性。
- 独特性好，信息量丰富，适用于在海量特征数据库中进行快速、准确的匹配。
- 多量性，即使少数的几个物体也可以产生大量 SIFT 特征向量。

- 速度相对较快，经过优化的 SIFT 匹配算法甚至可以达到实时的要求。
- 可扩展性强，可以很方便地与其他形式的特征向量进行联合。

13.2.1　基本方法与实现策略

本节方法实现的总体流程图见图 13.7。

实现的具体步骤如下：

(1) 读入 SAR 图像。

读入同一地区不同时相的两幅已配准和校正的 SAR 图像 $\boldsymbol{I}$ 和 $\boldsymbol{J}$。

(2) 归一化。

按照下式，对 SAR 图像 $\boldsymbol{I}$ 和 $\boldsymbol{J}$ 进行归一化，得到归一化后的 SAR 图像：

$$\boldsymbol{I}' = \frac{\boldsymbol{I} - \min(\boldsymbol{I})}{\max(\boldsymbol{I})} \tag{13.2}$$

$$\boldsymbol{J}' = \frac{\boldsymbol{J} - \min(\boldsymbol{J})}{\max(\boldsymbol{J})} \tag{13.3}$$

其中 $\boldsymbol{I}'$表示 SAR 图像 $\boldsymbol{I}$ 归一化后的 SAR 图像，min(·)表示取最小值操作，max(·)表示取最大值操作，$\boldsymbol{J}'$表示 SAR 图像 $\boldsymbol{J}$ 归一化后的 SAR 图像。

(3) 构造训练特征。

采用平移不变特征变换 SIFT 方法，分别提取两幅归一化后 SAR 图像 $\boldsymbol{I}'$和 $\boldsymbol{J}'$的平移不变特征变换 SIFT 特征 $\boldsymbol{S}_1$ 和 $\boldsymbol{S}_2$。对两组平移不变特征变换 SIFT 特征 $\boldsymbol{S}_1$ 和 $\boldsymbol{S}_2$ 进行级联操作，得到级联后的特征 $\boldsymbol{S}$。对级联后的特征 $\boldsymbol{S}$，采用主成分分析 PCA 算法进行降维，得到降维后的特征 $\boldsymbol{S}'$。

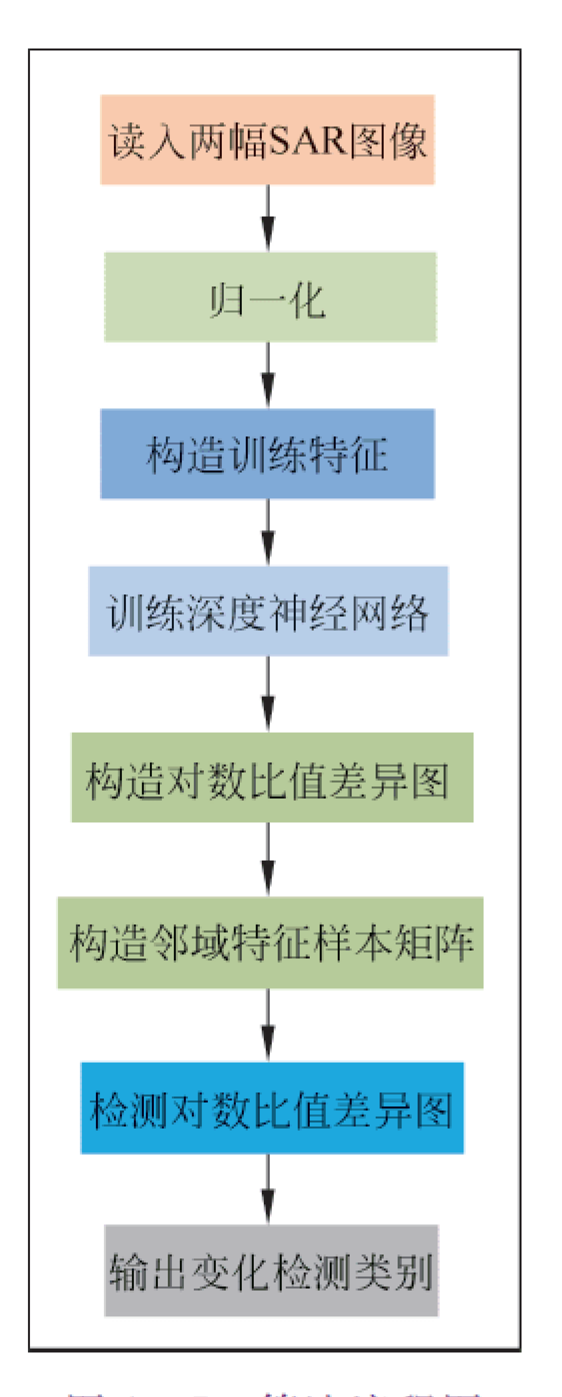


图 13.7　算法流程图

(4) 将降维后的特征 $\boldsymbol{S}'$输入到深度神经网络中，训练深度神经网络。训练深度神经网络的具体操作步骤如下：

① 初始化受限玻尔兹曼机(RBM)的参数。

② 待训练的特征 $\boldsymbol{S}'$使用受限玻尔兹曼机(RBM)进行训练，得到权重和偏置，网络层数设为 4 个隐藏层，每一层节点数目分别为 250、150、100、2。深度神经网络的每个隐藏层为一个受限玻尔兹曼机(RBM)，每一层训练 50 代。

③ 使用基于最小交叉熵的 BP 神经网络对 RBM 训练网络进行微调，训练代数为 50 代。

④ 得到训练好的深度神经网络。

(5) 按照式(13.4)，计算读入的两幅 SAR 图像的对数比值差异图像：

$$\boldsymbol{D} = \left|\log\left(\frac{\boldsymbol{I}+1}{\boldsymbol{J}+1}\right)\right| \tag{13.4}$$

其中 $\boldsymbol{D}$ 表示读入的两幅 SAR 图像的对数比值差异图像，$\log(\cdot)$表示取自然对数操作，$|\cdot|$ 表示取绝对值操作，$\boldsymbol{I}$ 和 $\boldsymbol{J}$ 分别表示读入的 SAR 图像。

(6) 构造对数比值差异图像 $\boldsymbol{D}$ 的邻域特征样本矩阵。

采用邻域特征提取方法，从对数比值差异图像 $\boldsymbol{D}$ 构成的像素值矩阵中提取每个像素点的邻域特征向量。

邻域特征提取方法的具体操作步骤如下：①在对数比值差异图像 $\boldsymbol{D}$ 上选取一个大小为 $n\times n$ 像素的滑动窗口，将所选窗口所有像素点的值拉成一个 $1\times N$ 维的特征向量，其中 n 为滑动窗口的大小，$N=n\times n$；②从左到右、从上到下依次滑动窗口，得到对数比值差异图像 $\boldsymbol{D}$ 上所有像素点的邻域特征向量。

将对数比值差异图像 $\boldsymbol{D}$ 所有像素点的邻域特征向量组成一个 $M\times N$ 维的邻域特征样本矩阵，其中 M 表示对数比值差异图像 $\boldsymbol{D}$ 中所有像素点的总数，N 表示对数比值差异图像 $\boldsymbol{D}$ 中每个像素点的邻域特征向量的维数。

(7) 检测对数比值差异图像 $\boldsymbol{D}$。

将对数比值差异图像 $\boldsymbol{D}$ 的邻域特征样本矩阵输入到训练好的深度神经网络中，检测对数比值差异图像 $\boldsymbol{D}$，得到对数比值差异图像 $\boldsymbol{D}$ 中每个像素检测为变化类或非变化类的检测类别。

(8) 输出检测类别。

此方法与现有其他技术相比具有以下优点：

- 由于此方法采用了尺度不变特征变换 SIFT 算法，提取了读入 SAR 图像的 SIFT 特征，并利用该特征对深度神经网络进行训练，克服了现有方法中训练样本的选取不可靠的问题，使得本发明提高了 SAR 图像变化检测的精度。
- 由于此方法提取了读入 SAR 图像的 SIFT 特征，该特征可以反映图像的局部特征，而且对仿射变换和噪声也保持一定程度的稳定性，克服了现有方法中受噪声影响导致不能有效检测出变化区域的问题，使得本发明提高了 SAR 图像变化检测的精度。
- 由于此方法提取了读入 SAR 图像的 SIFT 特征，该特征对图像旋转、尺度缩放以及亮度变化均具有不变性，因而对不同的图像的特征提取具有一定程度上的稳定性，克服了现有方法中对于不同的 SAR 图像变化检测鲁棒性不高的问题，使得本发明对于不同的 SAR 图像信息具有更强的适应性。

13.2.2 对比试验结果分析

下面结合仿真实验对本节方法的效果做进一步的说明。

1. 仿真条件

本节方法的仿真实验是在主频 2.30GHz 的 Intel Pentium(R) Dual-Core CPU、内存 5GB 的硬件环境和 MATLAB R2015a 的软件环境下进行的。

本节方法仿真实验所使用的仿真参数如下：

漏检数：统计实验结果图中发生变化区域的像素个数，与参考图中变化区域的像素个数进行对比，把参考图中发生变化但实验结果图中检测为未变化的像素个数，称为漏检数 FN。

误检数：统计实验结果图中未发生变化区域的像素个数，与参考图中未变化区域的像素个数进行对比，把参考图中未发生变化但实验结果图中检测为变化的像素个数，称为误检数 FP。

正确率 PCC：PCC=1－总错误数/总像素数。

衡量检测结果图与参考图一致性的 Kappa 系数：

$$\text{Kappa} = \frac{(\text{PCC} - \text{PRE})}{(1 - \text{PRE})} \tag{13.5}$$

其中正确率 PCC 表示实际的一致率，PRE 表示理论的一致率。

2. 仿真内容与结果分析

该方法的仿真实验采用广义 KI 阈值 GKI 方法、模糊局部信息 C 均值聚类 FLICM 方法作为对比方法，分别对三组 SAR 图像数据包括 Bern 地区、Ottawa 地区和黄河入海口地区进行变化检测的检测结果进行对比。

以下具体对各组实验的结果进行分析：

第一组 SAR 图像数据及相应的变化检测参考图是 Bern 地区的 SAR 图像，如图 13.8 所示，图像大小为 301×301，图 13.8(a)是 1999 年 4 月 Bern 地区的 SAR 图像，图 13.8(b)是 1999 年 5 月 Bern 地区的 SAR 图像，图 13.8(c)是 Bern 地区相应的变化检测参考图。

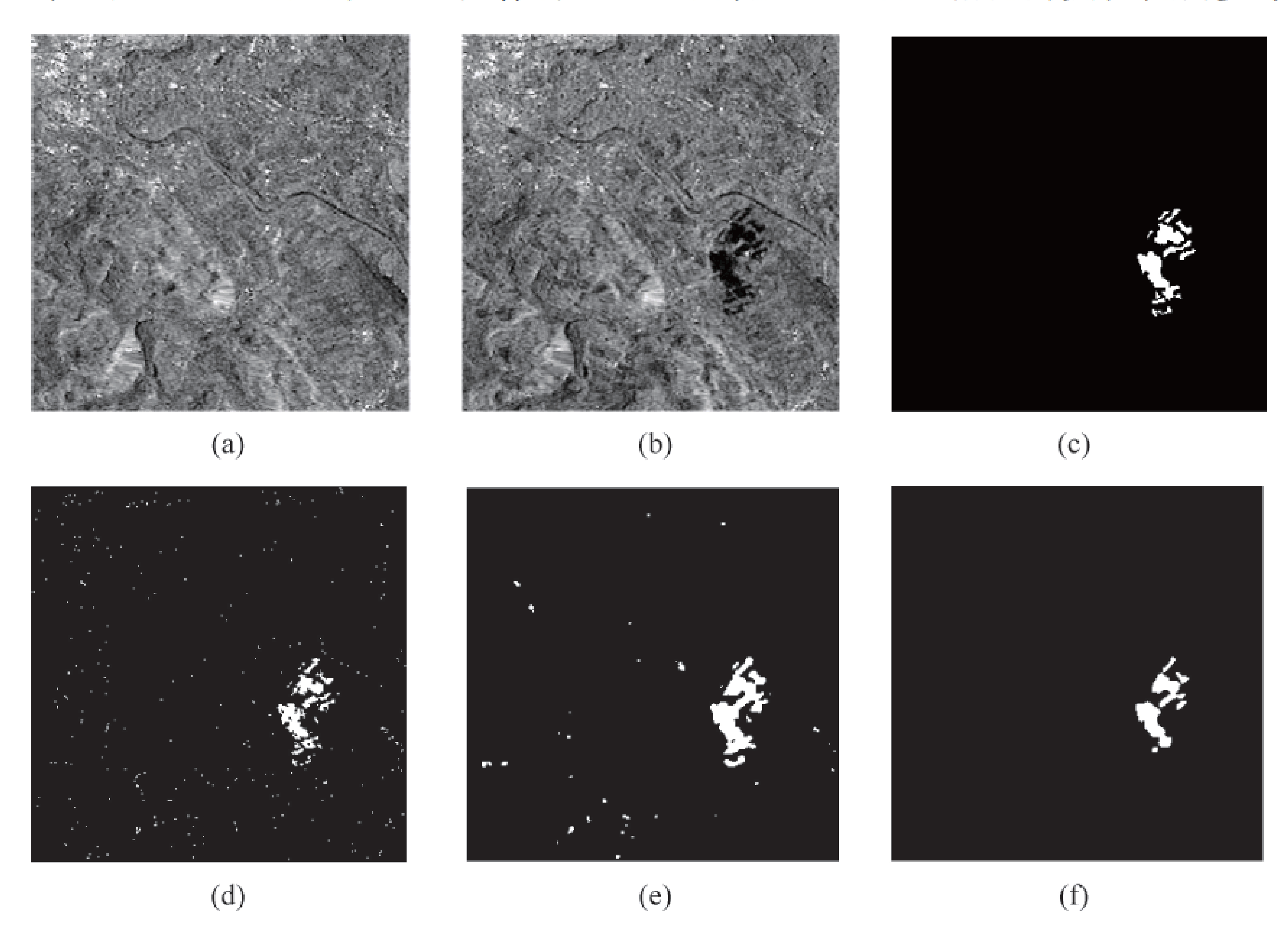


图 13.8　Bern 地区的 SAR 图像变化检测结果

仿真实验的分析：采用广义 KI 阈值 GKI 方法、模糊局部信息 C 均值聚类 FLICM 方法和采用本节方法得到的变化检测结果分别如图 13.8(d)～图 13.8(f)所示，对检测结果的具体对比分析见表 13.2。从图 13.8 的视觉效果可以看出，采用本发明的检测结果图与参考图最为接近。由表 13.2 可以看出，此方法错检的像素数分别比 GKI 和 FLICM 少了 401 个和 497 个，而总的错误像素数也分别比两者少了 183 个和 278 个，Kappa 系数也比二者分别高 2.69％和 5.36％。

表 13.2　Bern 地区变化检测结果

方　法	FN	FP	FN+FP	PCC/％	Kappa/％
GKI	56	513	569	99.37	79.13
FLICM	55	609	664	99.27	76.46
本节方法	274	112	386	99.57	81.82

第二组 SAR 图像数据及相应的变化检测参考图是 Ottawa 地区的 SAR 图像，如图 13.9 所示，图像大小为 290×350，图 13.9(a)是 1997 年 5 月 Ottawa 地区的 SAR 图像，图 13.9(b)是 1997 年 8 月 Ottawa 地区的 SAR 图像，图 13.9(c)是 Ottawa 地区相应的变化检测参考图。

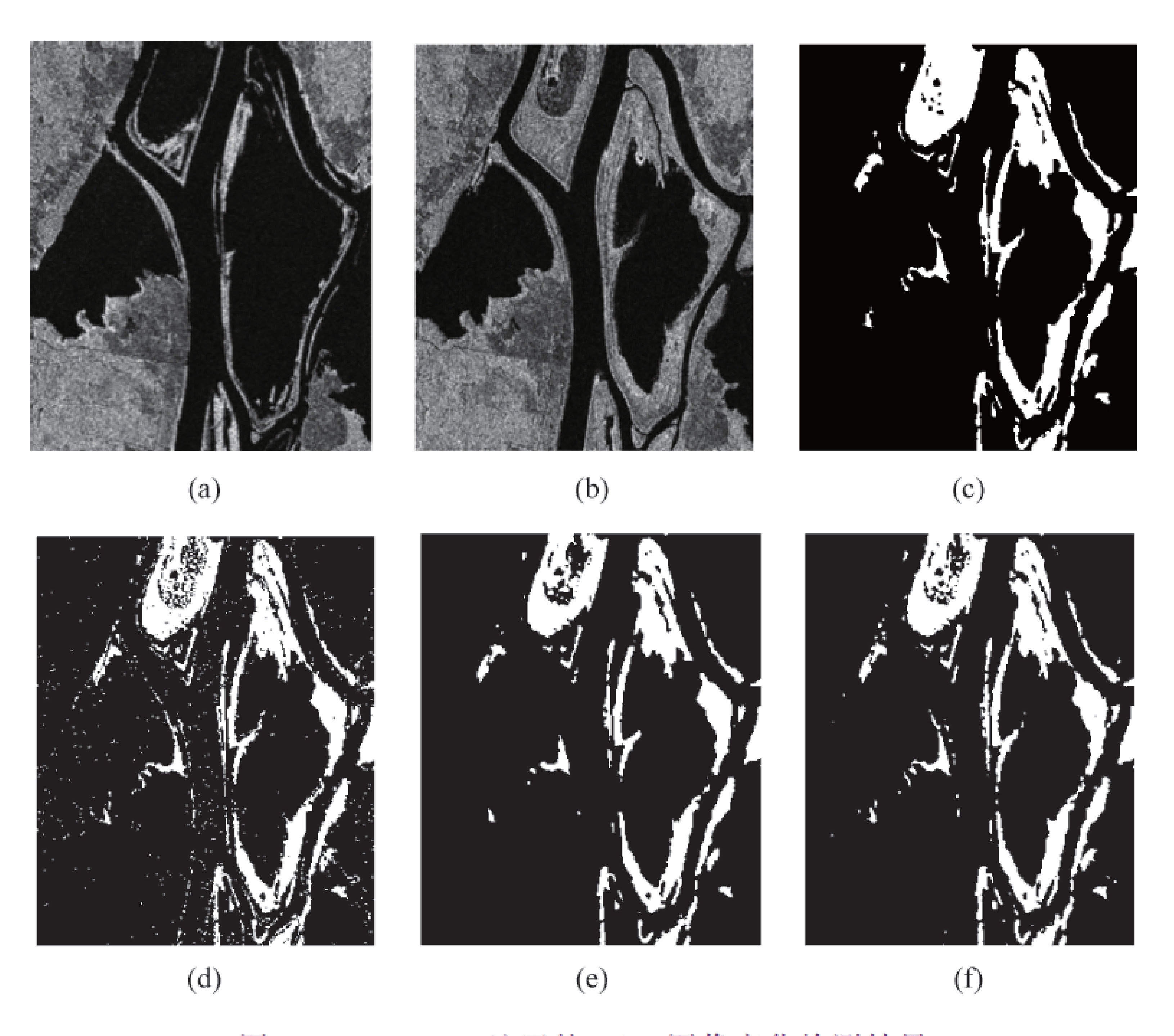


图 13.9　Ottawa 地区的 SAR 图像变化检测结果

仿真实验的分析：采用广义 KI 阈值 GKI 方法、模糊局部信息 C 均值聚类 FLICM 方法和采用本节方法得到的变化检测结果分别如图 13.9(d)～图 13.9(f)所示，对检测结果的具体对比分析见表 13.3。从图 13.9 的视觉效果可以看出，采用本发明的检测结果图与参考图最为接近。由表 13.3 可以看出，本节方法的错检像素数比 GKI 少了 904 个，漏检的像素数分别比 GKI 和 FLICM 少了 1814 个和 1493 个，而总的错误像素数分别比两者少了 3003 个和 415 个，Kappa 系数也比二者分别高 11.00%和 3.98%。

表 13.3　Ottawa 地区变化检测结果

方　　法	FN	FP	FN+FP	PCC/%	Kappa/%
GKI	2962	2391	5353	94.73	80.29
FLICM	2641	124	2765	97.28	87.31
本节方法	1148	1202	2350	97.68	91.29

第三组 SAR 图像数据及相应的变化检测参考图是黄河口农田地区的 SAR 图像，如图 13.10 所示，图像大小为 306×291，图 13.10(a)是 2008 年 6 月黄河口农田地区的 SAR 图像，图 13.10(b)是 2009 年 6 月黄河口农田地区的 SAR 图像，图 13.10(c)是黄河口农田地区相应的变化检测参考图。

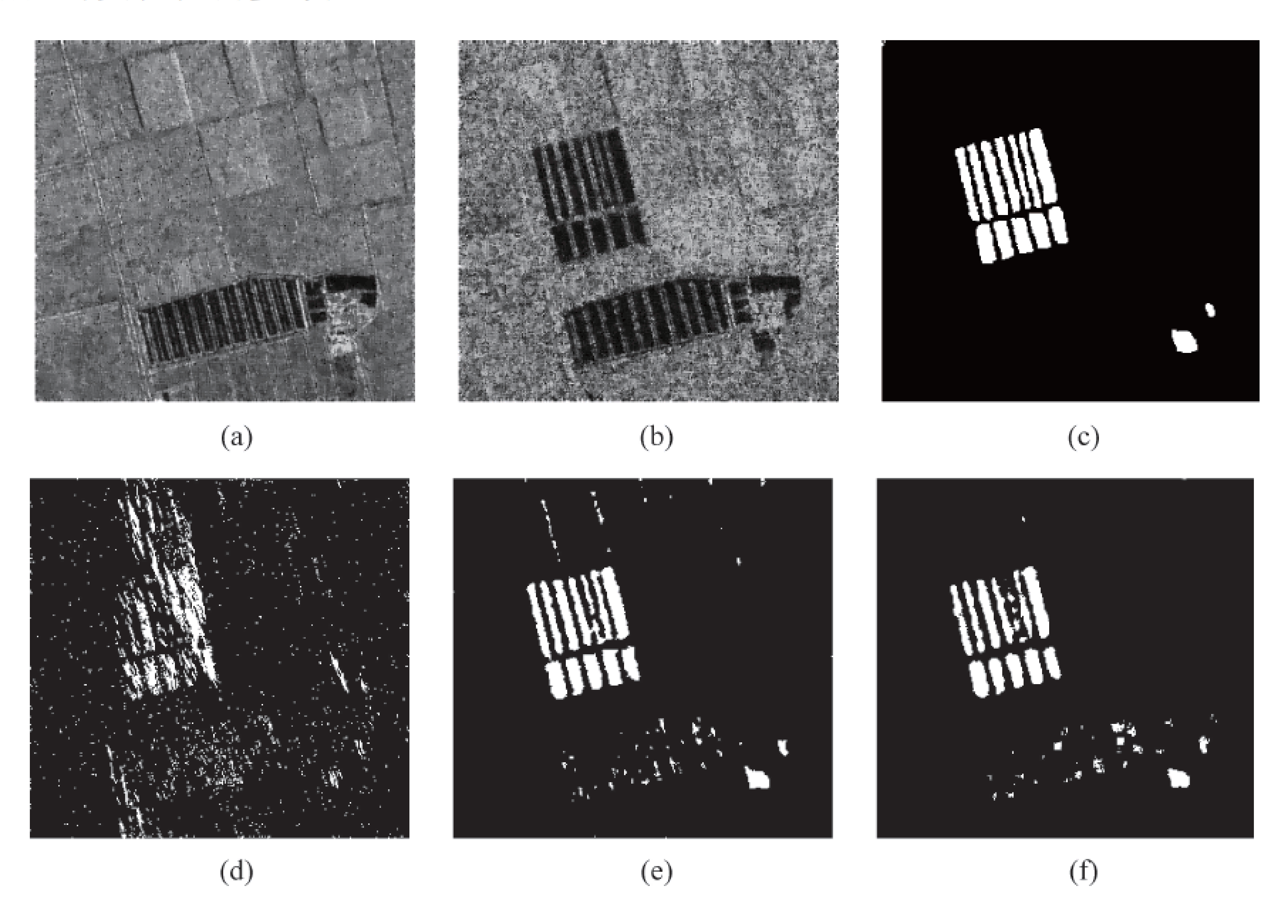


图 13.10　黄河口农田地区的 SAR 图像变化检测结果

仿真实验的分析：采用广义 KI 阈值 GKI 方法、模糊局部信息 C 均值聚类 FLICM 方法和采用本发明方法得到的变化检测结果分别如图 13.10(d)～图 13.10(f)所示，对检测结果

的具体对比分析见表13.4。从图13.10的视觉效果可以看出，采用本发明的检测结果图与参考图最为接近。由表13.4可以看出，本节方法的错检像素数分别比GKI和FLICM少了2392个和30个，漏检的像素数分别比GKI和FLICM少了2037个和88个，而总的错误像素数分别比两者少了4429个和118个，Kappa系数也比二者分别高44.27%和0.79%。

表13.4　黄河口农田地区变化检测结果

方　法	FN	FP	FN+FP	PCC/%	Kappa/%
GKI	2988	2836	5824	93.46	41.00
FLICM	1039	474	1513	98.30	84.48
本节方法	951	444	1395	98.43	85.27

13.3　基于SAE的SAR图像变化检测

本节介绍一种基于堆栈自编码(SAE)的SAR图像变化检测方法，以实现对SAR图像变化区域的准确检测。该方法先对图像进行FCM聚类得到预分类图，并将粗略的变化检测结果用以指导后面Softmax分类器微调网络的训练。首先各从两幅图中选取对应位置的样本，拉成向量，再将其级联起来作为网络的输入。网络分为两部分，第一部分即是逐层训练SAE网络，第二部分是将训练好的各层再串联起来，加上Softmax分类器，形成微调分类器网络，此网络可以看成是一个二分类网络。样本被分为变化或不变化两类中的一类，即得到最终的变化检测结果。

这里先简要介绍一下上面用的FCM聚类。

模糊C均值(Fuzzy C-means)算法简称FCM算法，是一种基于目标函数的模糊聚类算法，主要用于数据的聚类分析。理论成熟，应用广泛，是一种优秀的聚类算法。FCM算法是一种基于划分的聚类算法，它的思想就是使得被划分到同一簇的对象之间相似度最大，而不同簇之间的相似度最小。FCM算法是普通C均值算法的改进，普通C均值算法对于数据的划分是硬性的，而FCM则是一种柔性的模糊划分。

FCM算法的一般步骤为：

(1) 确定分类数，指数m的值，确定迭代次数(这是结束的条件，当然结束的条件可以有多种)；

(2) 初始化一个隶属度U(注意条件——和为1)；

(3) 根据U计算聚类中心C；

(4) 这个时候可以计算目标函数J了；

(5) 根据C返回去计算U，回到步骤(3)，一直循环直到结束。

13.3.1　基本方法与实现策略

本节方法实现的总体流程图如图13.11所示。

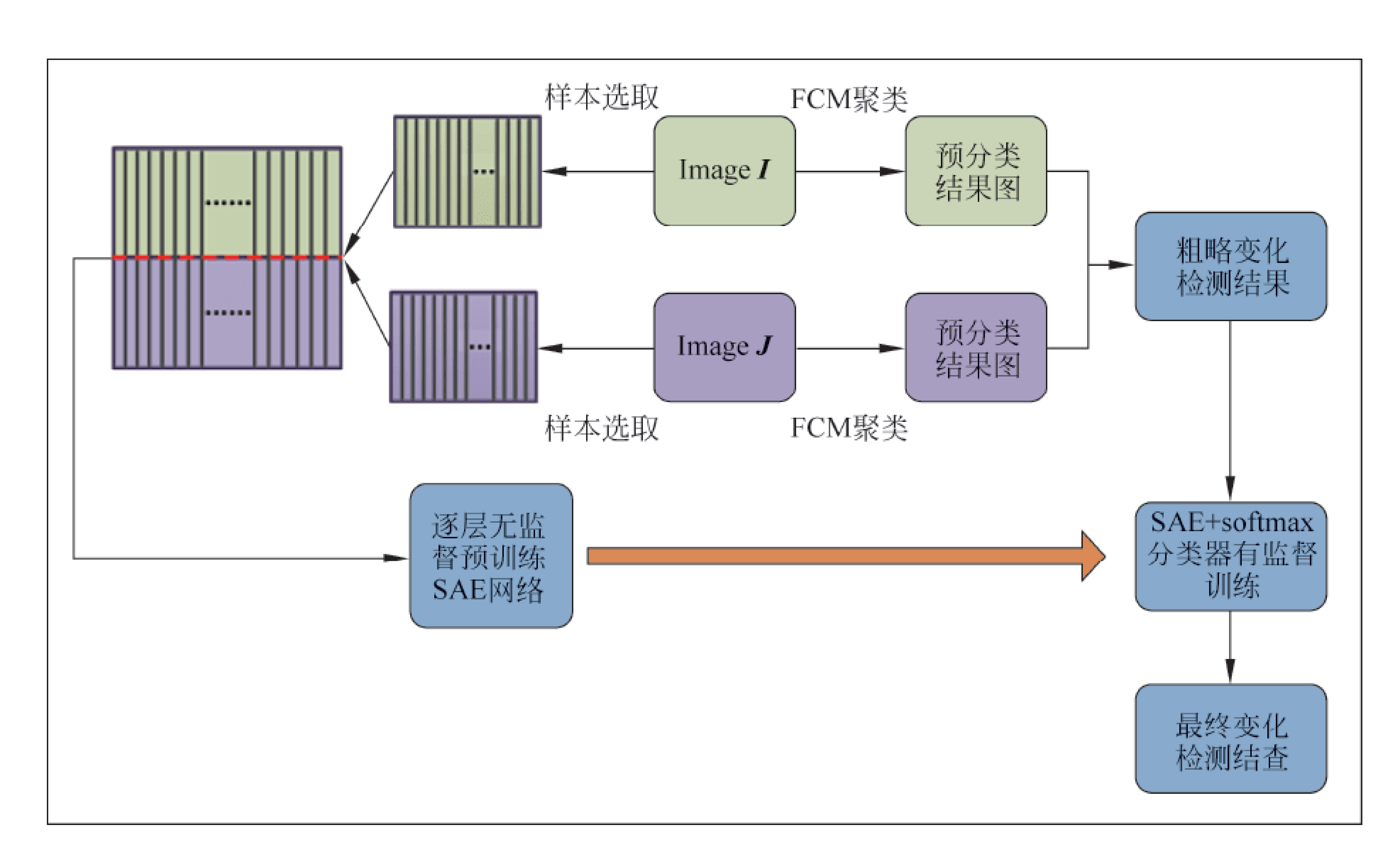


图 13.11　本节算法总流程图

整体步骤分为以下 6 步。

(1) 读入 SAR 图像。

读入同一地区不同时相的两幅已配准和校正的 SAR 图像 $\boldsymbol{I}$ 和 $\boldsymbol{J}$。

(2) 归一化预处理。

按照式(13.6)，对 SAR 图像 $\boldsymbol{I}$ 和 $\boldsymbol{J}$ 进行归一化，得到归一化后的 SAR 图像：

$$\boldsymbol{I}' = \frac{\boldsymbol{I} - \min(\boldsymbol{I})}{\max(\boldsymbol{I}) - \min(\boldsymbol{I})} \tag{13.6}$$

$$\boldsymbol{J}' = \frac{\boldsymbol{J} - \min(\boldsymbol{J})}{\max(\boldsymbol{J}) - \min(\boldsymbol{J})} \tag{13.7}$$

其中 $\boldsymbol{I}'$表示 SAR 图像 $\boldsymbol{I}$ 归一化后的 SAR 图像，min(·)表示取最小值操作，max(·)表示取最大值操作，$\boldsymbol{J}'$表示 SAR 图像 $\boldsymbol{J}$ 归一化后的 SAR 图像。

(3) 粗略变化检测结果的生成。

利用 FCM 的聚类方法对两幅影像进行预分类，并对两个图像的分类结果对比，若两幅图类标一致，则为未变化类，否则为变化类，据此得到粗略变化检测结果，用做后面训练有监督网络的标签。通过 FCM 算法得到预分类结果图的流程如图 13.12 所示。

(4) 选择样本。

① 训练样本选取

训练样本的选取是根据粗略变化检测结果选择“纯净”样本作为训练样本，不同的是这里选择两个原始图像对应区域图块(5×5)分别拉成向量，然后级联起来作为训练数据，训练类标是由粗略检测结果所得到。

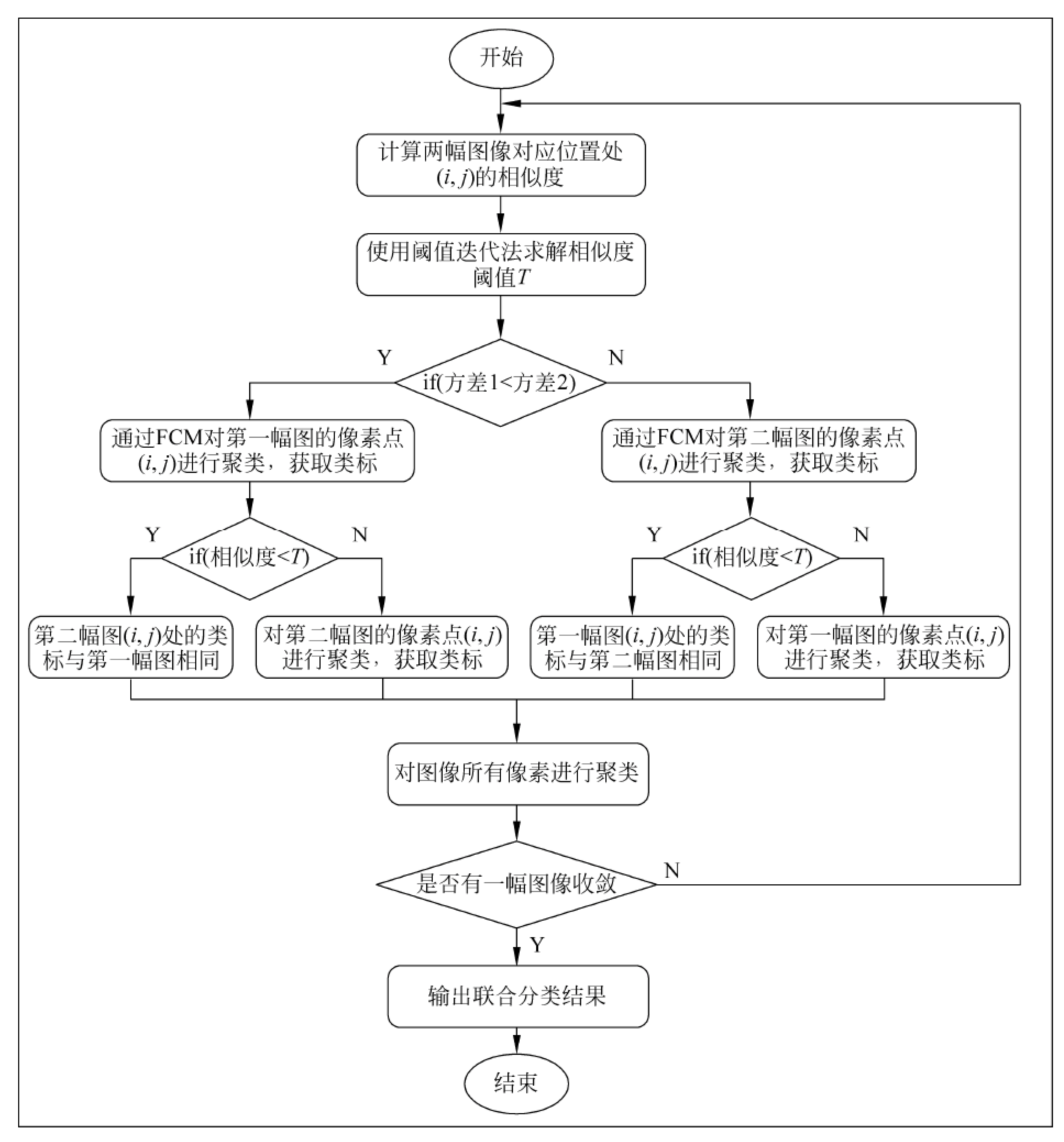


图 13.12　预分类结果图的生成

② 测试样本选取

测试样本选取同样不考虑块内类标纯度，通过滑窗的方式在整个图像上选取样本，并采取和训练样本相同的处理方式。

(5) 训练 SAE 网络。

自编码网络(SAE)是一种学习输入数据中隐含的一些特定结构的学习神经网络，由于自编码网络是无监督训练的，加之多层架构已经在许多分类和回归问题上取得了很好的结果。典型的自编码网络是应用反向传播算法使得网络的输出值最大限度趋近于输入值，网络的各个隐层单元即为输入数据的一种特征表示。栈式自编码神经网络是一个由多层稀疏

自编码器组成的神经网络，其前一层自编码器的输出作为其后一层自编码器的输入。假定用$\boldsymbol{W}^{(k,1)}$，$\boldsymbol{W}^{(k,2)}$，$\boldsymbol{b}^{(k,1)}$，$\boldsymbol{b}^{(k,2)}$表示第 k 个自编码器对应的$\boldsymbol{W}^{(1)}$，$\boldsymbol{W}^{(2)}$，$\boldsymbol{b}^{(1)}$，$\boldsymbol{b}^{(2)}$参数，那么该栈式自编码网络的编码过程就是按照从前往后的顺序执行每一层自编码器的编码步骤：

$$\boldsymbol{a}^{(l)} = f(\boldsymbol{z}^{(l)}) \tag{13.8}$$

$$\boldsymbol{z}^{(l+1)} = \boldsymbol{W}^{(l,1)} \cdot \boldsymbol{a}^{(l)} + \boldsymbol{b}^{(l,1)} \tag{13.9}$$

同理，栈式神经网络的解码过程就是按照从后往前的顺序执行每一层自编码器的解码步骤：

$$\boldsymbol{a}^{(n+l)} = f(\boldsymbol{z}^{(n+l)}) \tag{13.10}$$

$$\boldsymbol{z}^{(n+l+1)} = \boldsymbol{W}^{(n+l,1)} \cdot \boldsymbol{a}^{(n+l)} + \boldsymbol{b}^{(n+l)} \tag{13.11}$$

其中 $\boldsymbol{a}^{(n)}$ 是最深层隐藏单元的激活值，其包含了我们感兴趣的信息，这个向量也是对输入值的更高阶的表示。本实验采用栈式自编码(SAE)网络进行变化检测，该检测过程分为以下两个步骤：

① 无监督预训练

通过三层栈式自编码网络对原始图像进行特征提取，网络结构如图 13.13 所示：图中画出了三层 SAE 逐层训练过程，每一层都通过重构输入数据，并使重构数据尽可能与输入数据一致，即减少重构误差来训练本层网络。第二层将第一层的输出特征作为输入数据，同样最小化重构误差训练本层网络，以此类推，逐层训练每一层特征。如图 13.13 所示，这里输入层维度是 2×(5×5)维($\boldsymbol{x}$)，每一层的隐层节点为 100 维($\boldsymbol{X}_1$)——50 维($\boldsymbol{X}_2$)——25 维($\boldsymbol{X}_3$)。

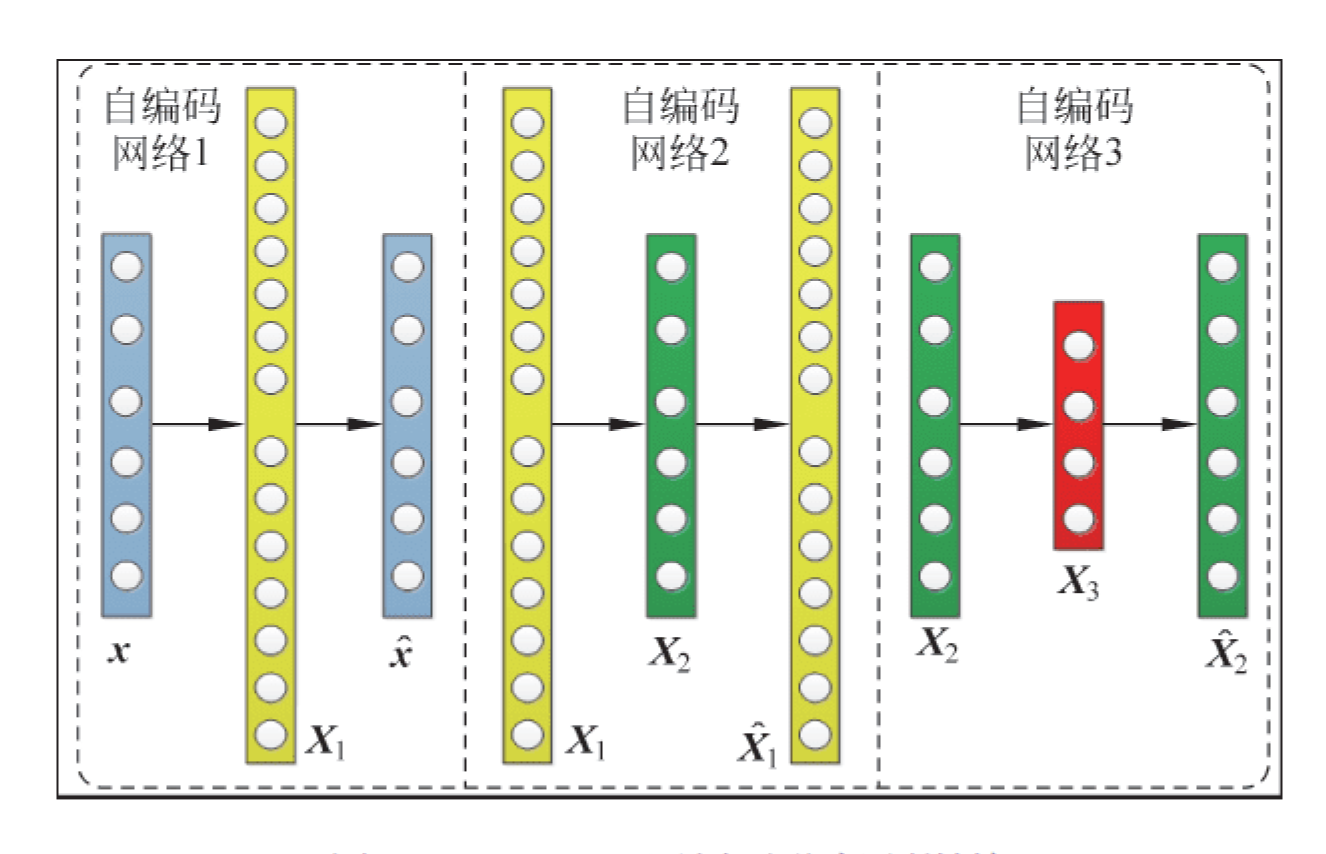


图 13.13　SAE 逐层无监督预训练

② 有监督微调分类器网络

构建一个神经网络模型如图 13.14 所示，使用(1)中训练好的每层特征初始化该网络模型，并对最深层的 25 维特征用 Softmax 分类器进行二分类，即最终的 Softmax 分类器输出是一个二维向量，表示样本变化的概率估计，1 表示完全变化，0 表示不变化。最终取两者较大值作为样本变化与否的判定。在步骤(4)中，我们已经选择好了一部分“纯净”样本，这些样本就可以用来微调这个分类器网络。微调过程采用 BP 算法。

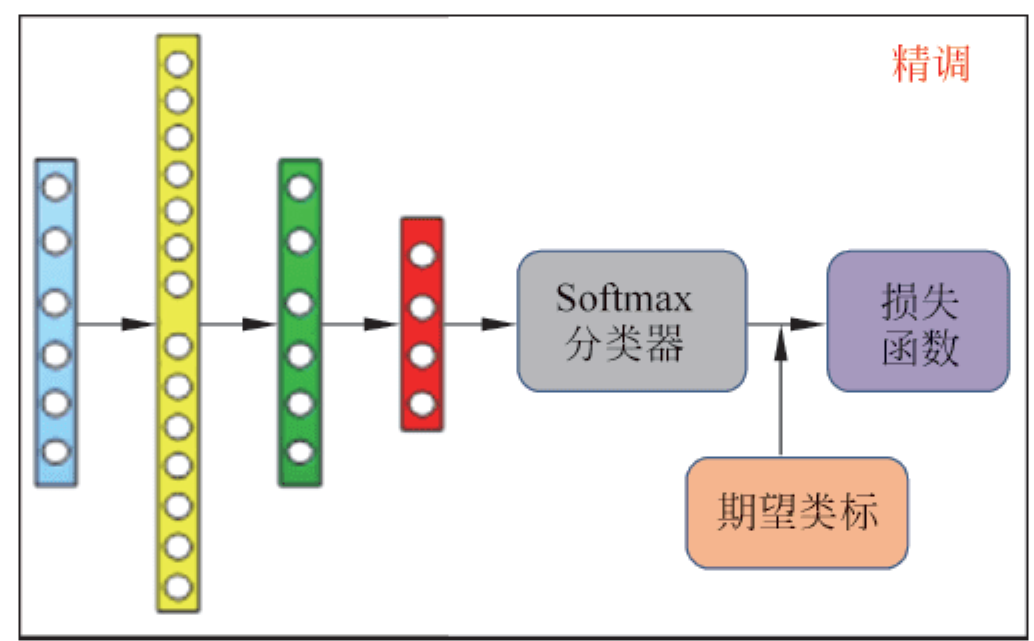


图 13.14 有监督精调分类器网络

(6) 通过网络最终的输出,判定样本所在区域最终的变化检测结果。

13.3.2 实验结果和分析

1. 仿真条件

本节方法的仿真实验是在主频 2.30GHz 的 Intel Pentium(R) Dual-Core CPU、内存 5GB 的硬件环境和 MATLAB R2015a 的软件环境下进行的。本节方法仿真实验所使用的仿真参数与 13.2 节相同,即漏检数 FN、误检数 FP、准确率 PCC 和 Kappa 系数。

2. 仿真结果

实验一:Ottawa 地区变化检测结果如图 13.15 所示。

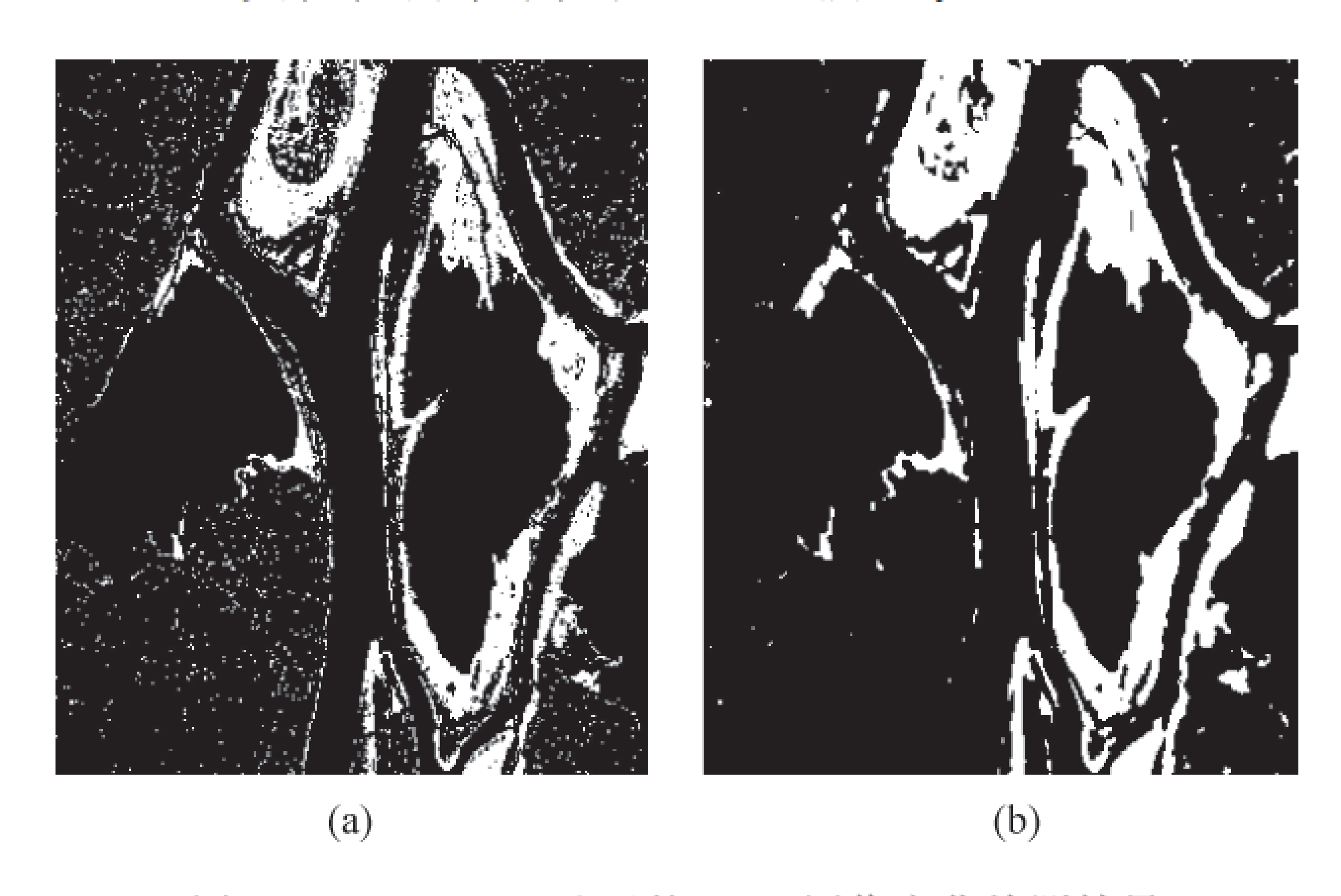
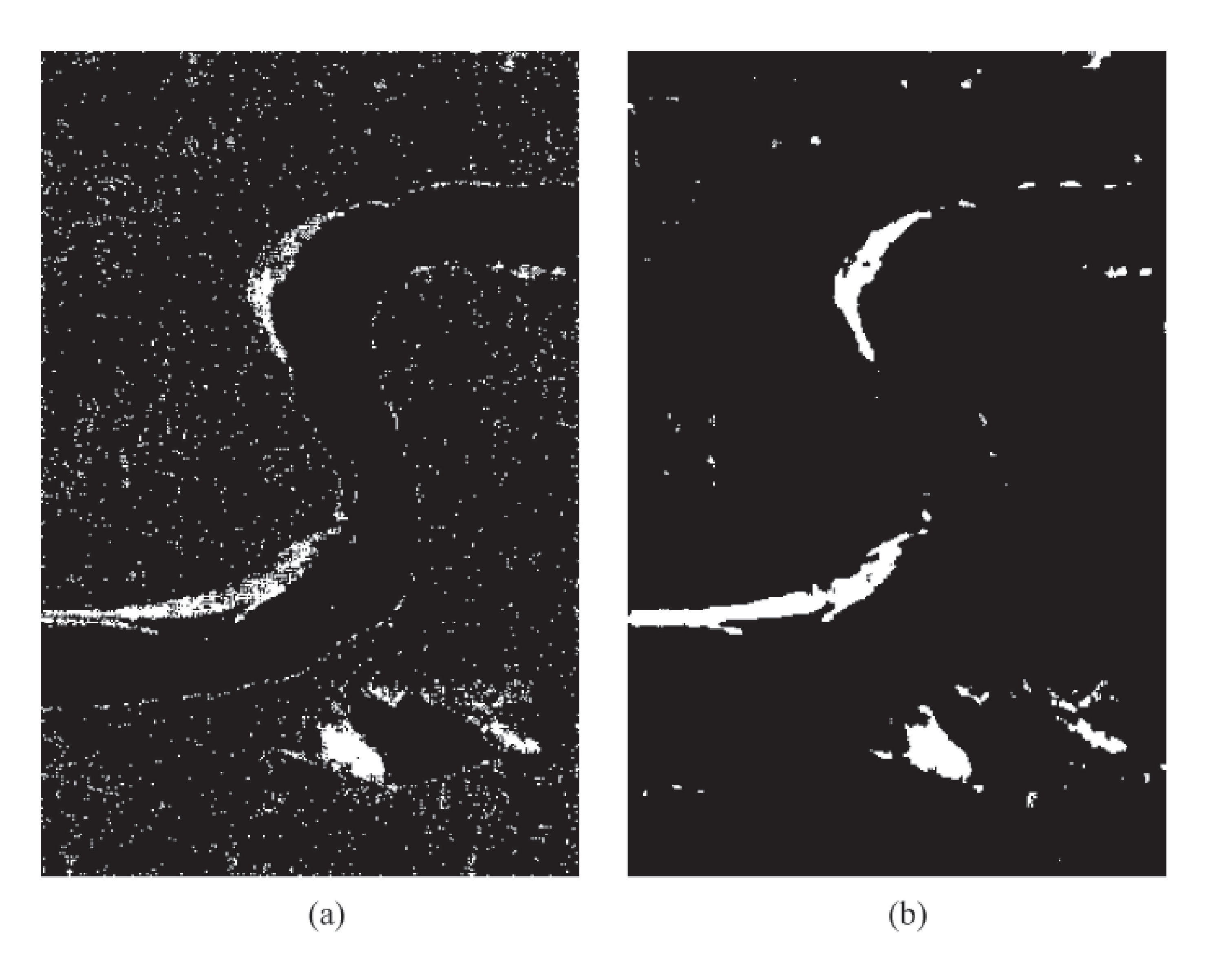


图 13.15 Ottawa 地区的 SAR 图像变化检测结果

结果分析:如图 13.15(a)所示为根据初始分类结果得到的变化区域估计,图 13.15(b)所示为通过 SAE 网络训练之后的变化检测结果。可以明显看出,SAE 训练有效减少了图

像噪声，对变化细节的检测更加精确。

实验二：黄河口 S 形地区的变化检测结果如图 13.16 所示。

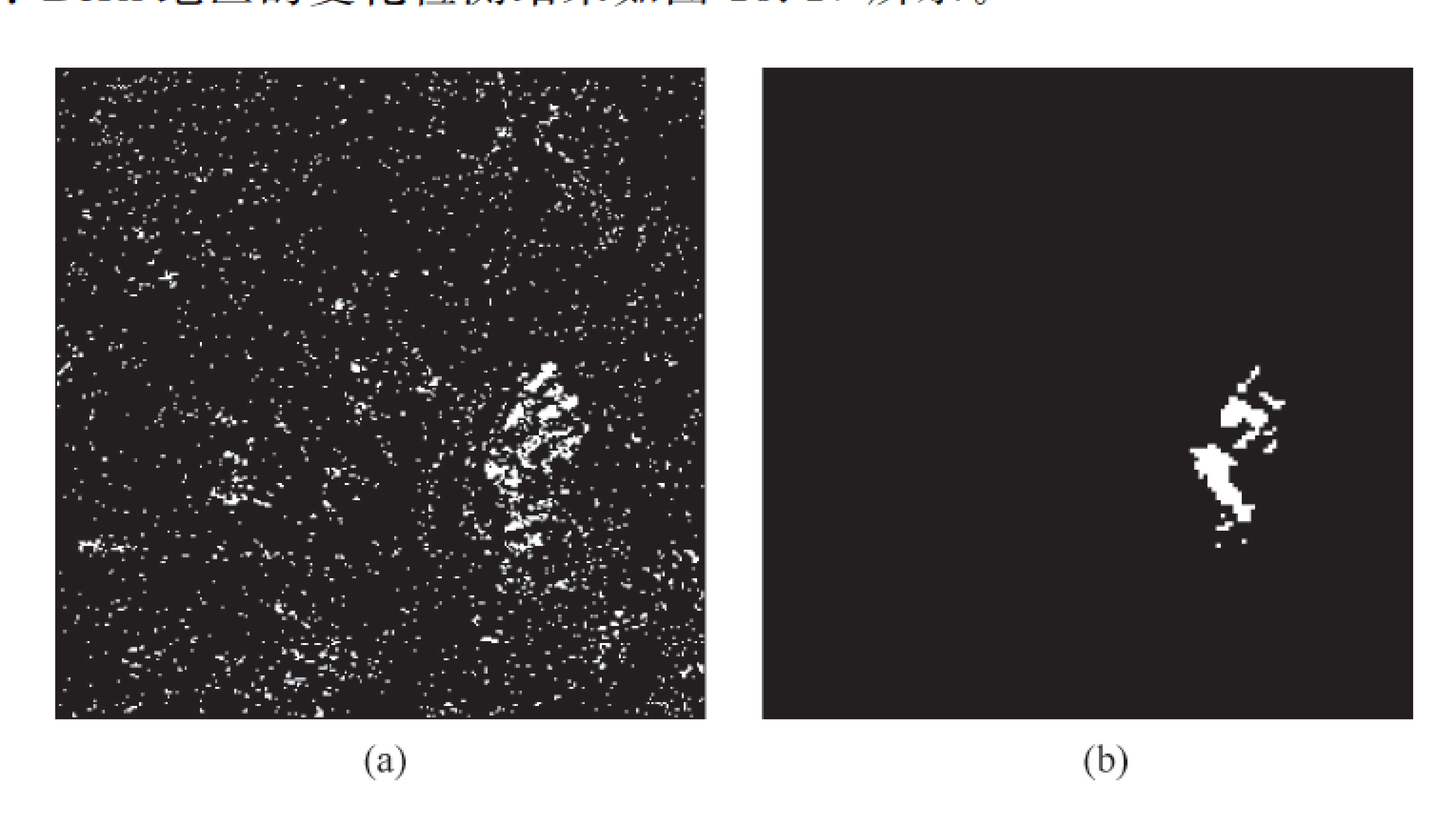


(a)　　(b)

图 13.16　黄河口数据集 S 形地区的 SAR 图像变化检测结果

结果分析：如图 13.16(a)所示为根据初始分类得到的变化区域估计，图 13.16(b)为最终的变化检测结果。从实验结果可以看出，最终变化检测结果避免了一部分噪声，仍然有一些噪声，观察这些噪声处在粗略变化检测结果中基本覆盖面积也比较大，推断出检测不精确的原因可能是由于选取样本纯度不够，或者由于噪声影响类标不正确。

实验三：Bern 地区的变化检测结果如图 13.17 所示。



(a)　　(b)

图 13.17　Bern 地区的 SAR 图像变化检测结果

结果分析：如图 13.17(a)所示为根据初始分类得到的变化区域估计，图 13.17(b)为最终的变化检测结果。该组实验检测结果对噪声避免较好，但是变化区域边缘检测有些缺陷，分析可能与所取的像素邻域大小有关，可以改变邻域大小测试，找到合适的邻域范围得到较好的变化检测结果。

性能分析：表 13.5 所示为三个地区在五个性能方面的评价，可以看出该组实验的 PCC 都比较高，但错误检测数仍然存在，可以按照以上每组实验的分析改进实验。

表 13.5　三组数据的变化检测结果

地　　区	FN	FP	FN+FP	PCC/%	Kappa/%
Ottawa	3085	252	3337	96.78	88.23
黄河口 S 形	984	989	1973	98.65	76.45
Bern	60	407	467	99.34	75.98

13.4　基于 CNN 的 SAR 图像变化检测

13.4.1　基本方法与实现策略

本节方法实现的总体流程图如图 13.18 所示。

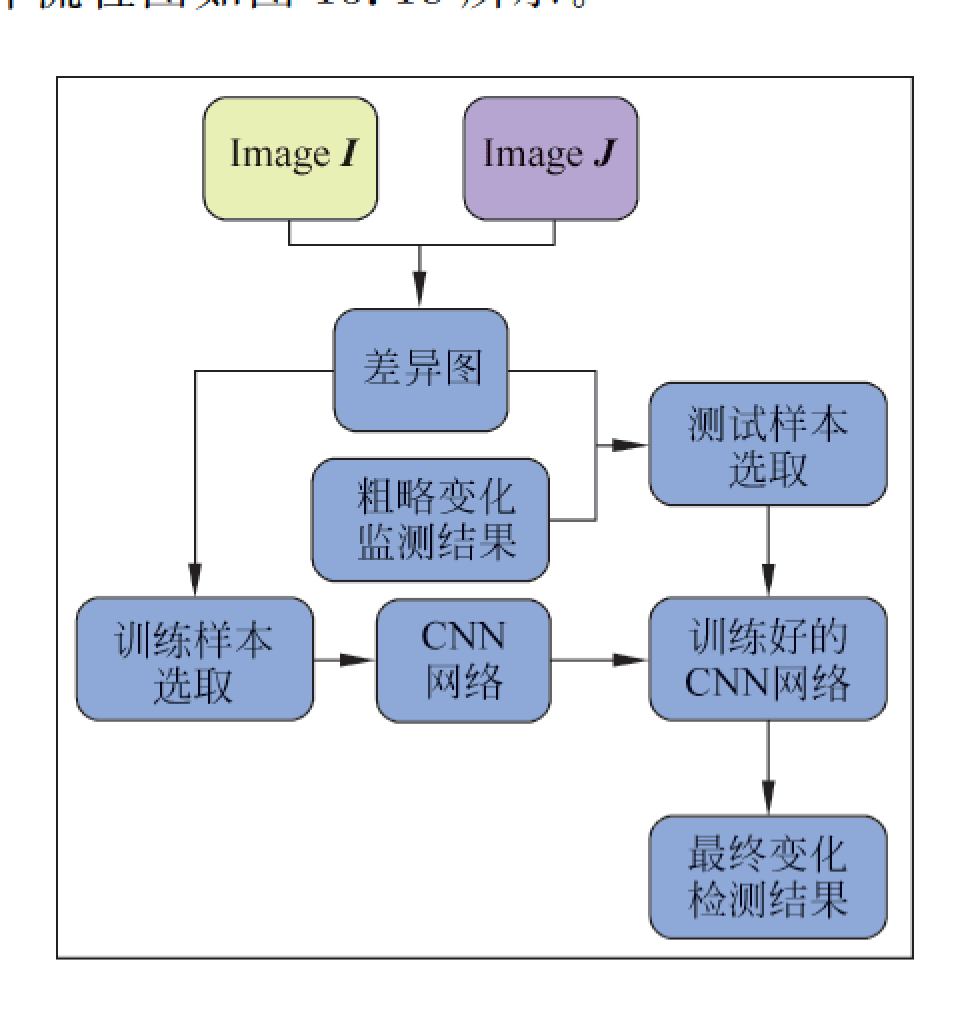


图 13.18　本节算法总流程图

整体步骤分为以下 6 步。

(1) 读入 SAR 图像。

读入同一地区不同时相的两幅已配准和校正的 SAR 图像 $\boldsymbol{I}$ 和 $\boldsymbol{J}$。

(2) 归一化预处理。

按照下式，对 SAR 图像 $\boldsymbol{I}$ 和 $\boldsymbol{J}$ 进行归一化，得到归一化后的 SAR 图像：

$$\boldsymbol{I}' = \frac{\boldsymbol{I} - \min(\boldsymbol{I})}{\max(\boldsymbol{I}) - \min(\boldsymbol{I})} \tag{13.12}$$

$$\boldsymbol{J}' = \frac{\boldsymbol{J} - \min(\boldsymbol{J})}{\max(\boldsymbol{J}) - \min(\boldsymbol{J})} \tag{13.13}$$

其中 $\boldsymbol{I}'$表示 SAR 图像 $\boldsymbol{I}$ 归一化后的 SAR 图像，min(·)表示取最小值操作，max(·)表示取最大值操作，$\boldsymbol{J}'$表示 SAR 图像 $\boldsymbol{J}$ 归一化后的 SAR 图像。

(3) 粗略变化检测结果的生成。

根据 SAR 影像变化检测的基本流程范式，即预处理、差异图生成、差异图分析的方法得到粗略的变化检测图。预处理让两幅影像在空域和谱域上具有一致可比性。生成差异图初步区分两幅 SAR 影像中未变化类和变化类，为后续的差异图分析环节提供基础。差异图分析对差异图分析生成一幅黑白二值图。

(4) 选择样本。

① 训练样本选取

训练样本选择时我们采用固定大小的窗口在差异图和粗略变化检测图上滑动提取我们需要的样本。训练样本选取时需要相对“纯净”的样本，如果样本点 p 满足以下公式，则该样本点可以作为训练样本。

$$\frac{N(p_{\xi\eta} \in N_{ij} \wedge \Omega_{\xi\eta} = \Omega_{ij})}{n \times n} > \alpha \tag{13.14}$$

其中 N_{ij} 是样本点 p 的邻域，Ω_{ij} 是(i,j)点的类标，$N(p_{\xi\eta} \in N_{ij} \wedge \Omega_{\xi\eta} = \Omega_{ij})$表示样本点 p 邻域 N_{ij} 内等于中心点 p 的类标的数目。参数 α 以控制选择的样本好坏。

② 测试样本选取

测试样本选取时采用与选择训练样本时相同的模式，不同的是测试样本选取时不考虑块内类标纯度，将整幅图像滑框全部选择，因此测试样本所得的标记即为我们所需要的检测结果。

(5) 训练 CNN 网络。

由于 SAR 图像成像特性，在单极化 SAR 影像变化检测领域最为突出的困难是对相干斑噪声影响的克服。卷积神经网络(CNN)在结构方面具有局部感受野、权值共享等特性，充分利用了图像的邻域信息，保证了图像的位移、缩放、扭曲不变性。本实验采用卷积神经网络对差异图特征提取和分类。网络结构如图 13.19 所示。

该网络中应用了两个卷积层和一个下采样层，对提取到的特征利用 Softmax 分类器进行分类。第一层卷积层 conv1 用的是 3×3@10 的卷积核，type 类型为 same，即卷积前和卷积后尺寸相同；第二层卷积层 conv2 也用的是 3×3@10 的卷积核，type 类型为 same；池化层 pooling 的池化窗口为 2×2，type 类型为 max-pooling，即在每个 2×2 窗口中选择最大值作为输出。最后经过 Softmax 分类器，和上一节 SAE 的输出相同，转化成一个二分类问题，输出一个二维向量，通过和标签构成误差函数，反向回传并微调整个网络。

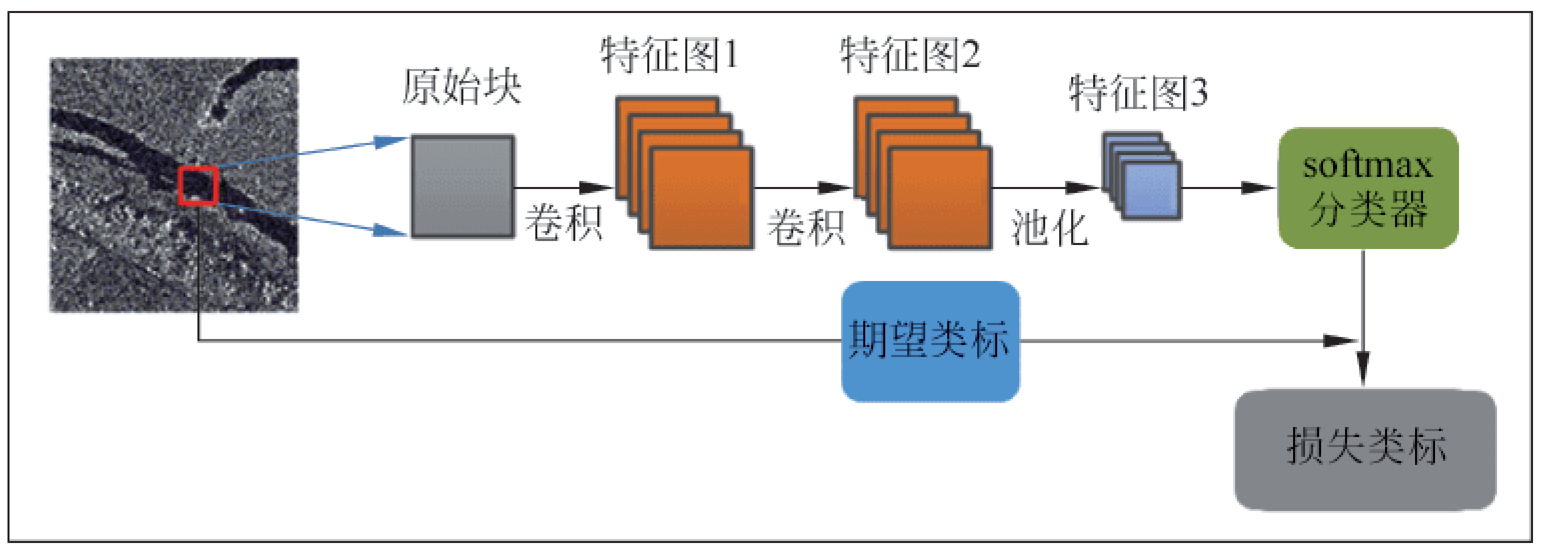


图 13.19 CNN 训练网络

（6）通过网络最终的输出，判定样本所在区域最终的变化检测结果。

13.4.2 对比试验结果分析

1. 仿真条件

本节方法的仿真实验是在主频 2.30GHz 的 Intel Pentium（R） Dual-Core CPU、内存 5GB 的硬件环境和 MATLAB R2015a 的软件环境下进行的。本节方法仿真实验所使用的仿真参数与 13.2 节相同，即漏检数 FN、误检数 FP、准确率 PCC 和 Kappa 系数。

2. 仿真结果

实验一：本实验中分别选取了 5×5、7×7、9×9 大小的图像块对 Ottawa 地区数据集进行实验。

结果分析：如图 13.20 所示，图中分别为 Ottawa 地区按照块大小分别为 5×5、7×7、9×9 所得的变化检测结果图，从图中可以看出当选取块大小为 7 时，得到的结果相对较好，因此之后的实验中我们选取大小为 7×7 的块。在该实验中我们从粗略变化检测图中选取训练样本后，根据训练样本的类标在两类样本中分别取一部分，作为训练数据训练网络，训练准确率达到 98%；所有样本为测试数据，测试准确率为 96%。

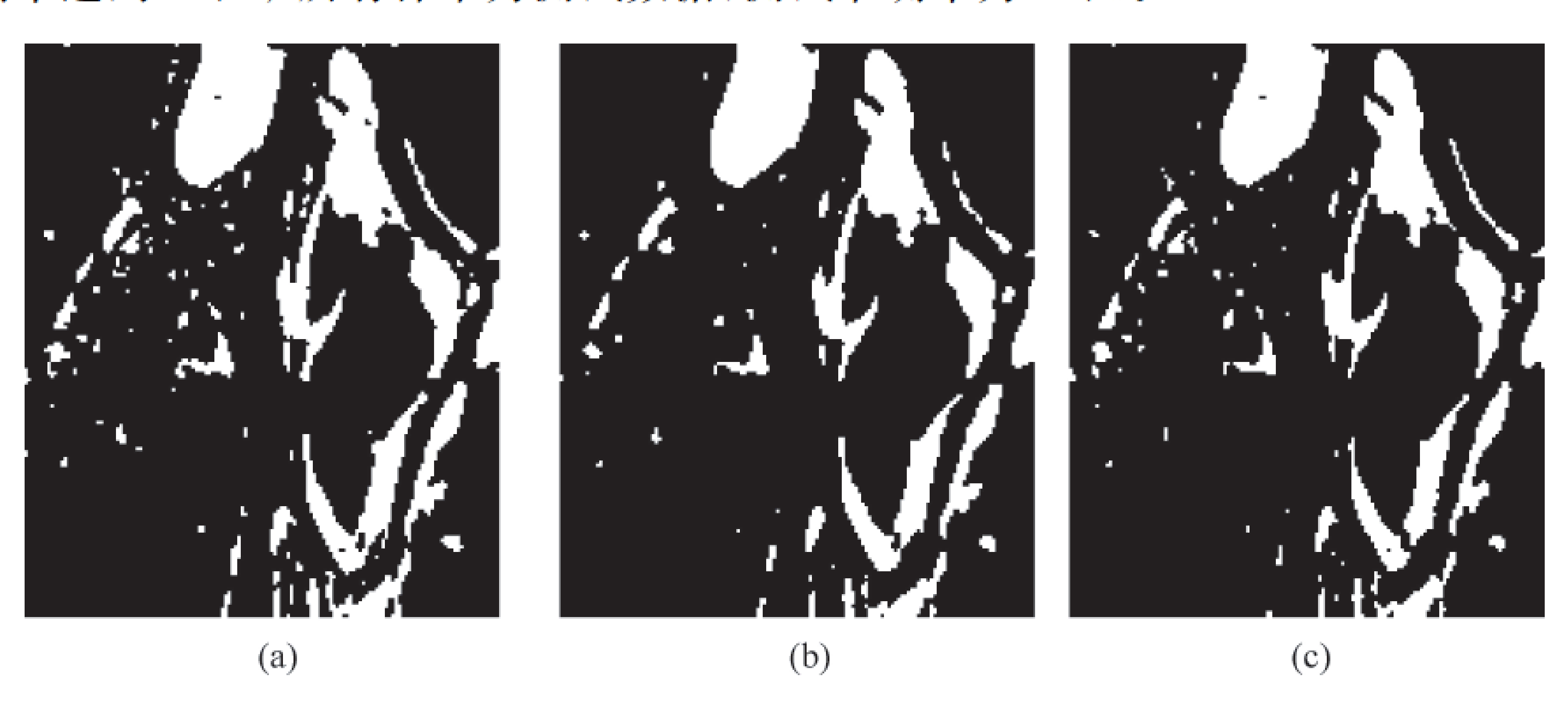
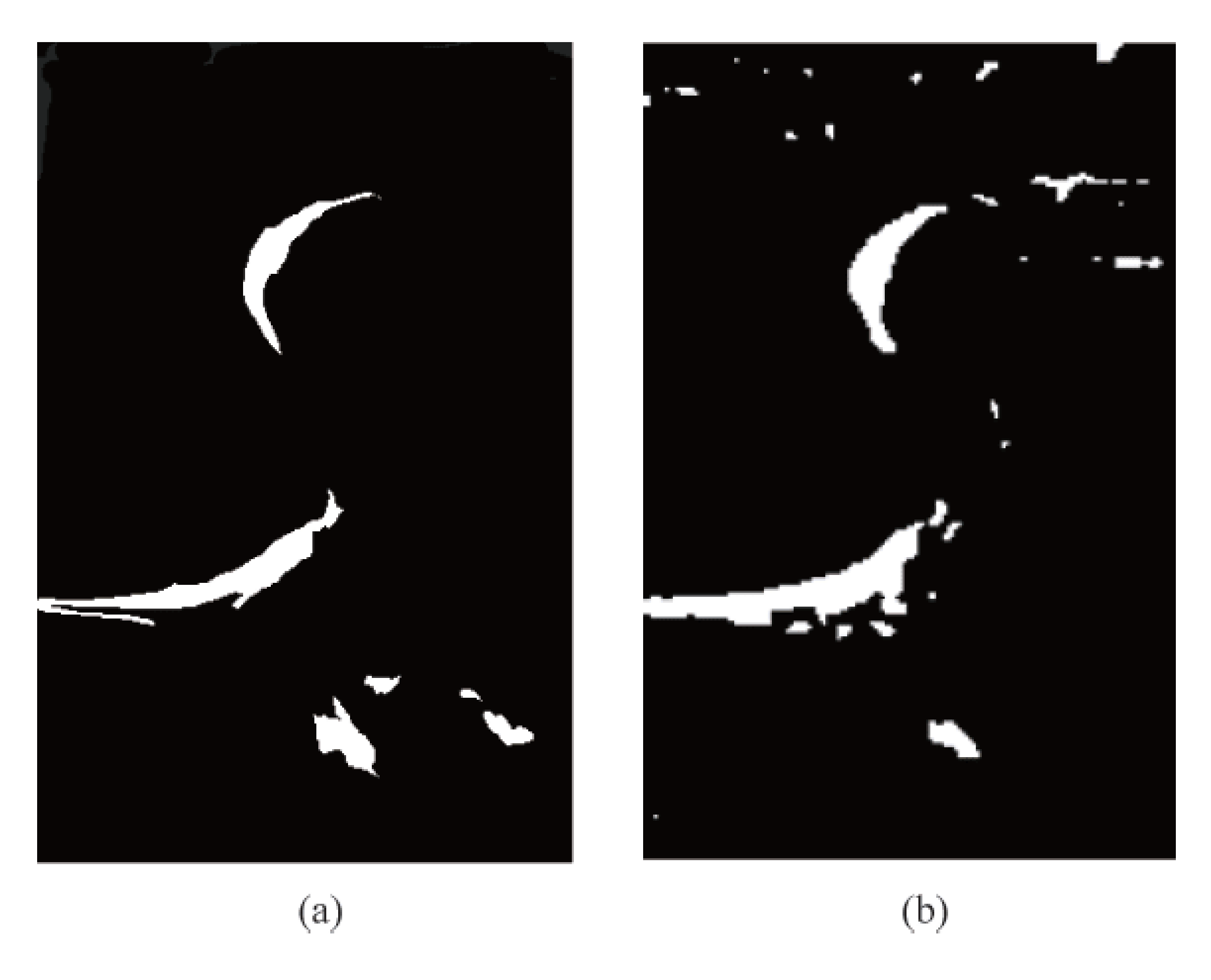


图 13.20 Ottawa 地区变化检测结果

实验二：黄河口 S 形地区变化检测结果如图 13.21 所示。

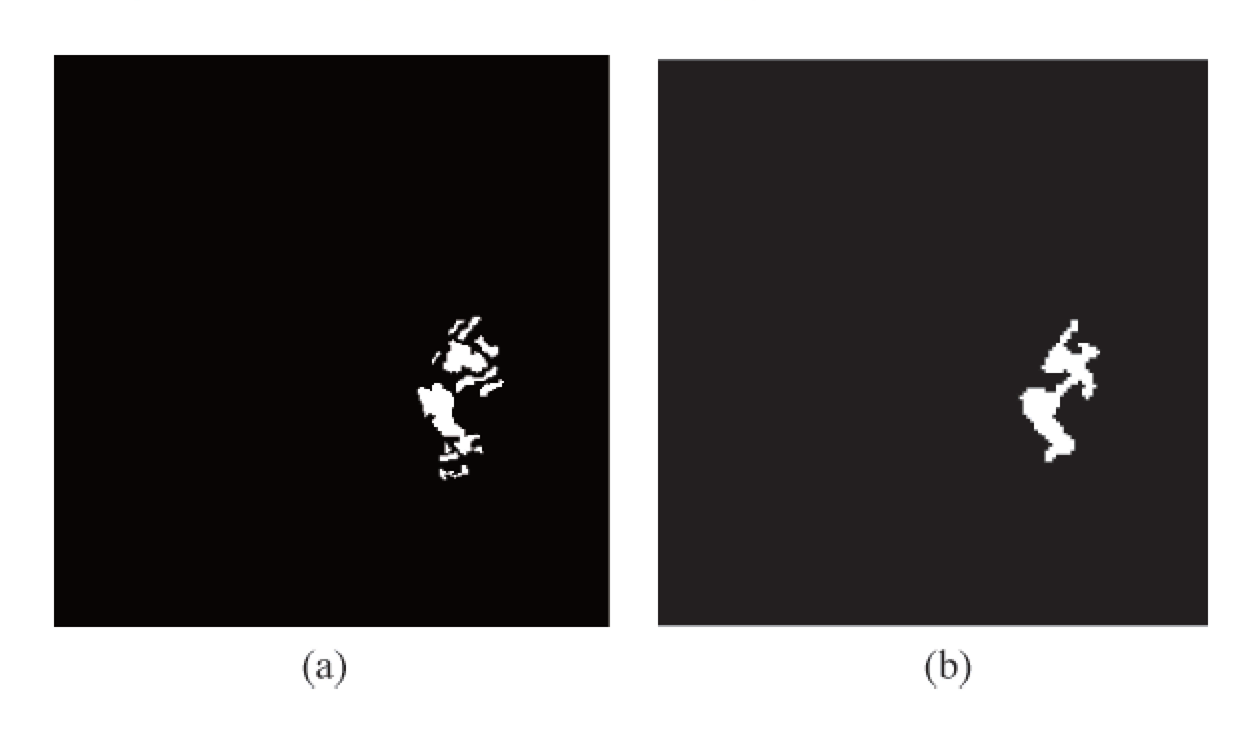


(a)　　(b)

图 13.21　黄河口 S 形地区变化检测结果

结果分析：依据 Ottawa 地区实践经验，我们知道选择 7×7 大小的块检测效果较好，因此，黄河流域数据集，我们也采用 7×7 大小的块做检测；如图 13.21(b)所示，图 13.21(a)为人工标记参考图。该检测结果基本将变化检测区域全部检测出来，但是仍然有一些噪声，分析可能的原因是粗略变化检测结果不好引起，因此，可以尝试改变粗略检测的方法，以进一步提高检测精度。

实验三：Bern 地区变化检测结果如图 13.22 所示。



(a)　　(b)

图 13.22　Bern 地区变化检测结果

结果分析：如图 13.22 所示为 bern 地区选取的图块尺寸为 7×7 时所得的变化检测结果图。该组实验结果有效地避免了噪声的干扰，变化区域的检测较完整，但对比参考图，仍然有一些细节检测不准确。由此可以推断出，取块的 CNN 检测方法可以适用于变化区域较大的变化检测算法。

性能分析：如表 13.6 所示为三组实验相应的性能指标，其中 PCC 指标即正确检测率都达到了 95%以上，但是 Kappa 系数还有待提高。

表 13.6　三组数据的变化检测结果

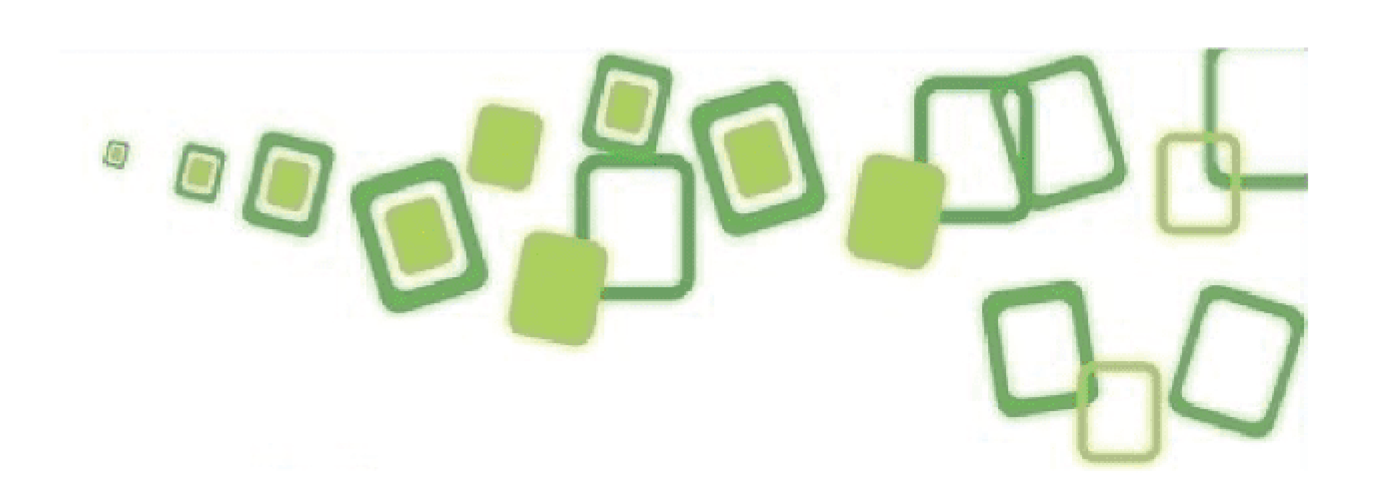
地　　区	FN	FP	FN+FP	PCC/%	Kappa/%
Ottawa	1926	2051	3977	96.32	85.91
黄河口 S 形	815	2596	3411	97.28	55.34
Bern	242	273	515	99.34	77.47

参考文献

[13.1] 罗永波，高贵. SAR 图像变化检测[C]. 全国信号和智能信息处理与应用学术会议专刊. 2015.
[13.2] 张辉. SAR 图像变化检测技术研究[D]. 电子科技大学，2008.
[13.3] 王娜，张景发. SAR 图像变化检测技术方法综述[J]. 地壳构造与地壳应力文集，2016.
[13.4] Hussain M，Chen D，Cheng A，et al. Change detection from remotely sensed images From pixel-based to object-based approaches[J]. Isprs Journal of Photogrammetry & Remote Sensing，2013，80(2)：91-106.
[13.5] Rignot EJM，Zyl JJV. Change detection techniques for ERS-1 SAR data[J]. IEEE Transactions on Geoscience & Remote Sensing，1993，31(4)：896-906.
[13.6] Zhong Y，Liu W，Zhao J，et al. Change Detection Based on Pulse-Coupled Neural Networks and the NMI Feature for High Spatial Resolution Remote Sensing Imagery[J]. IEEE Geoscience & Remote Sensing Letters，2015，12(3)：537-541.
[13.7] Zhang L，Zhang L，Du B. Deep Learning for Remote Sensing Data：A Technical Tutorial on the State of the Art[J]. IEEE Geoscience & Remote Sensing Magazine，2016，4(2)：22-40.
[13.8] Lecun Y，Bengio Y，Hinton G. Deep learning[J]. Nature，2015，521(7553)：436-44.
[13.9] Hinton G E，Osindero S，Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation，2006，18(7)：1527-1554.
[13.10] Krizhevsky A，Sutskever I，Hinton G E. ImageNet classification with deep convolutional neural networks[J]. Advances in Neural Information Processing Systems，2012，25(2)：2012.
[13.11] Bengio Y，Guyon G，Dror V，et al. Deep learning of representations for unsupervised and transfer learning[J]. Workshop on Unsupervised & Transfer Learning，2011，7.
[13.12] 李玉峰，李景芳. 基于 FCM 算法的 SAR 图像相干斑噪声滤波算法研究[J]. 计算机应用研究，2016，33(8)：2496-2499.
[13.13] 全斯农，崔莹，熊博莅，等. 利用多尺度融合的 SAR 图像变化检测方法[J]. 信号处理，2016，32(4)：430-437.
[13.14] 田淞，宋建社，张雄美，等. KM-SVM 法的 SAR 图像无监督变化检测[J]. 系统工程与电子技术，2015，37(5)：1042-1046.
[13.15] 郝睿，徐俊峰，王庆宝，等. 基于 BP 神经网络的多特征融合变化检测方法[J]. 海洋测绘，2016，36(1)：79-82.
[13.16] 辛芳芳，焦李成，王桂婷. 基于 Memetic 算法的 SAR 图像变化检测[J]. 红外与毫米波学报，

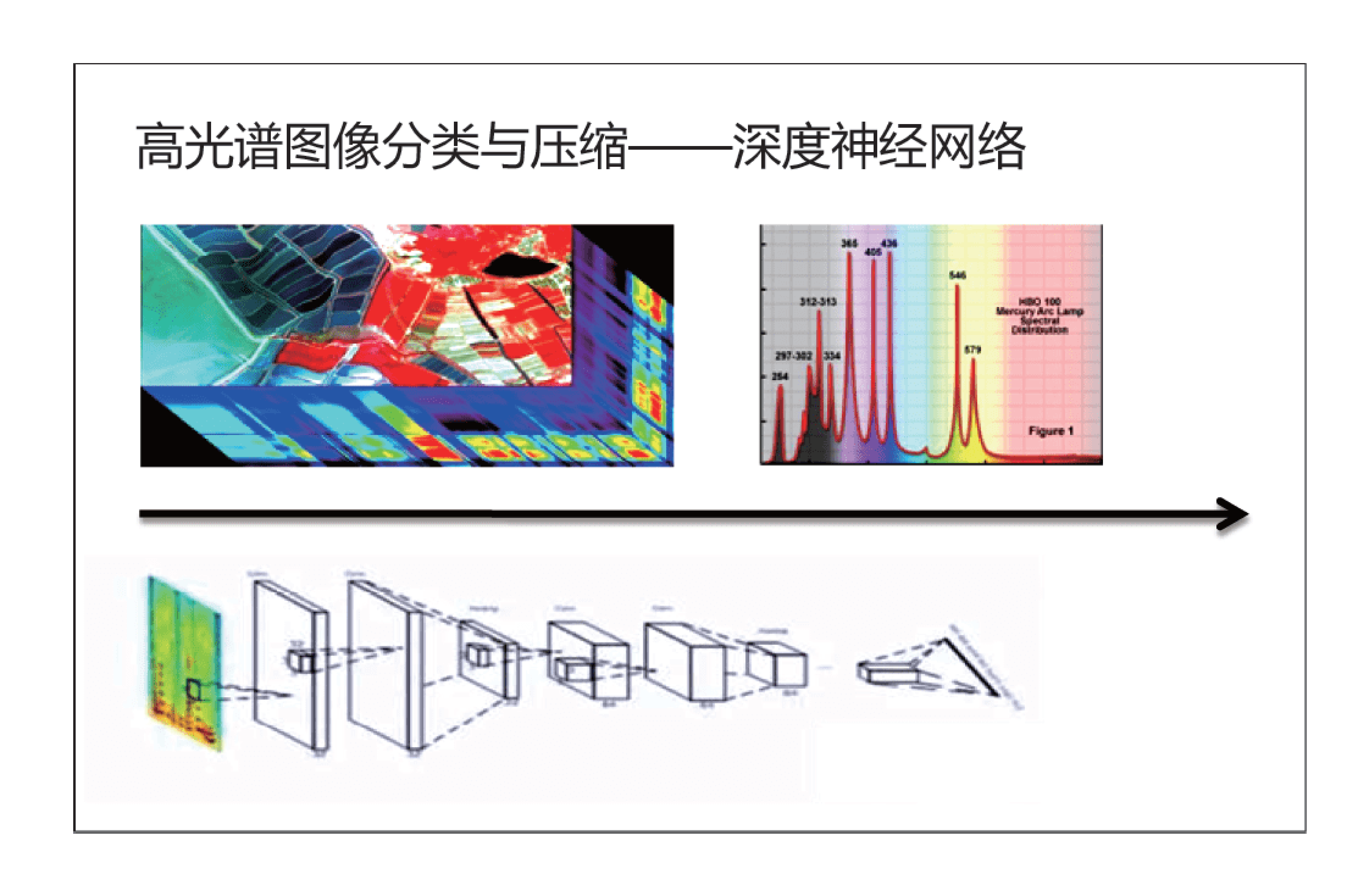
2012，31(1)：67-72.
[13.17] 辛芳芳，焦李成，王桂婷，等. 基于小波域 Fisher 分类器的 SAR 图像变化检测[J]. 红外与毫米波学报，2011，30(2)：173-178.
[13.18] 刘逸，慕彩红，刘敬. 结合邻域信息粒子群聚类用于 SAR 图像变化检测[J]. 西安电子科技大学学报，2015，42(1)：187-193.
[13.19] 庄会富，邓喀中，范洪冬. 纹理特征向量与最大化熵法相结合的 SAR 影像非监督变化检测[J]. 测绘学报，2016，45(3)：339-346.
[13.20] Lowe D G. Distinctive Image Features from Scale-Invariant Keypoints[J]. International Journal of Computer Vision，2004，60(2)：91-110.
[13.21] 李玉峰，李景芳. 基于图像配准的混合遗传 FCM 算法研究[J]. 电视技术，2016，40(3)：5-10.
[13.22] Liu R，Jia Z，Qin X，et al. SAR Image Change Detection Method Based on Pulse-Coupled Neural Network[J]. Journal of the Indian Society of Remote Sensing，2016，44(3)：443-450.
[13.23] Hrúz M，Kunešová M. Convolutional Neural Network in the Task of Speaker Change Detection [M]. Speech and Computer. Springer International Publishing，2016.
[13.24] Lyu H，Lu H，Mou L. Learning a Transferable Change Rule from a Recurrent Neural Network for Land Cover Change Detection[J]. Remote Sensing，2016，8(6)：506.
[13.25] Fang W，Hu R，Xu X，et al. A novel road network change detection algorithm based on floating car tracking data[J]. Telecommunication Systems，2016：1-7.
[13.26] Amin A M E，Liu Q，Wang Y. Convolutional neural network features based change detection in satellite images[C]. International Workshop on Pattern Recognition. International Society for Optics and Photonics，2016：100110W.
[13.27] Han X，Leung T，Jia Y，et al. MatchNet：Unifying feature and metric learning for patch-based matching[C]. Computer Vision and Pattern Recognition. 2015：3279-3286.
[13.28] Zagoruyko S，Komodakis N. Learning to compare image patches via convolutional neural networks[J]. 2015：4353-4361.
[13.29] Simoserra E，Trulls E，Ferraz L，et al. Discriminative Learning of Deep Convolutional Feature Point Descriptors[C]. IEEE International Conference on Computer Vision. IEEE，2015：118-126.
[13.30] Zbontar J，Lecun Y. Computing the Stereo Matching Cost with a Convolutional Neural Network [J]. 2014：1592-1599.
[13.31] Luo W，Schwing A G，Urtasun R，et al. Efficient Deep Learning for Stereo Matching[C]. computer vision and pattern recognition，2016：5695-5703.
[13.32] 辛芳芳. 基于 Fisher 分类器和计算智能的遥感图像变化检测[D]. 西安电子科技大学，2011.
[13.33] 焦李成，侯彪，尚荣华，杨淑媛等. 智能 SAR 影像变化检测[M]. 北京：科学出版社，2017.
[13.34] 武杰. 基于素描模型和可控核函数的 SAR 图像相干斑抑制[D]. 西安电子科技大学，2015.
[13.35] 石程. 基于视觉稀疏表示和深度脊波网络的遥感图像融合及分类[D]. 西安电子科技大学，2016.
[13.36] 万红林. 无降斑预处理的两对相 SAR 图像变化检测方法研究[D]. 西安电子科技大学，2011.
[13.37] 刘赶超. 基于双噪声相似性模型的 SAR 图像变化检测[D]. 西安电子科技大学，2016.

第14章



基于深度神经网络的高光谱图像分类与压缩

CHAPTER 14



14.1　数据集及研究目的

14.1.1　高光谱遥感技术

遥感(Remote Sensing)技术从20世纪60年代至今,已经历五十多年的发展。作为一种远距非接触的对地观测技术,通过传感器探测和记录地物目标反射或者辐射出的电磁波,根据不同地物的不同辐射波长,获得相关的信息。然后,对所获取的光谱数据进行处理和分析,根据数据的属性以及规律进行定性或定量表示,实现地物目标的分类和识别。遥感技术以电磁辐射理论作为基础,同时涉及地理学、光谱学、地质学、物理与几何光学、电磁波理论等多种学科。由于遥感图像具有较高的实时性,地物信息丰富,覆盖的地物面积广,目前已经成功应用于环境检测、地质勘探、军事侦察、大气海洋检测、精细农业等众多领域。随着硬件技术和成像设备的快速发展以及应用需求的不断变化,遥感图像需要具有更高的光谱分辨率、空间分辨率以及更加丰富的地物信息,此时,遥感技术从宽波段成像转向窄波段成像发展,高光谱遥感由此而产生。

高光谱遥感技术利用成像光谱仪,得到几百个很窄的光谱波段数据,光谱范围从可见光到红外光,其光谱分辨率相较于传统遥感数据要高很多,而光谱分辨率越高,越有可能对地物进行准确的识别。高光谱遥感能够在对地物进行成像的同时,对每个空间像元进行光谱成像,由于高光谱遥感的光谱通道很多且各光谱之间具有连续性,使得每一个像元的光谱信息都能够形成一条平滑连续的曲线。因此,高光谱遥感数据同时包含了丰富的空间信息与光谱信息,将两者相叠加可以得到一个立方体来代表高光谱遥感数据,使其具有图谱合一的性质,其中两维表示空间位置信息,在二维空间的基础上多出一维光谱信息而得到三维的高光谱数据立方体。图14.1给出高光谱数据的三维结构示意图,可以看到,除了普通图像的两个空间维度,还有一位特征维度,每一个像素都是一个具有多维特征的量,每个像素对应一条连续的光谱曲线。

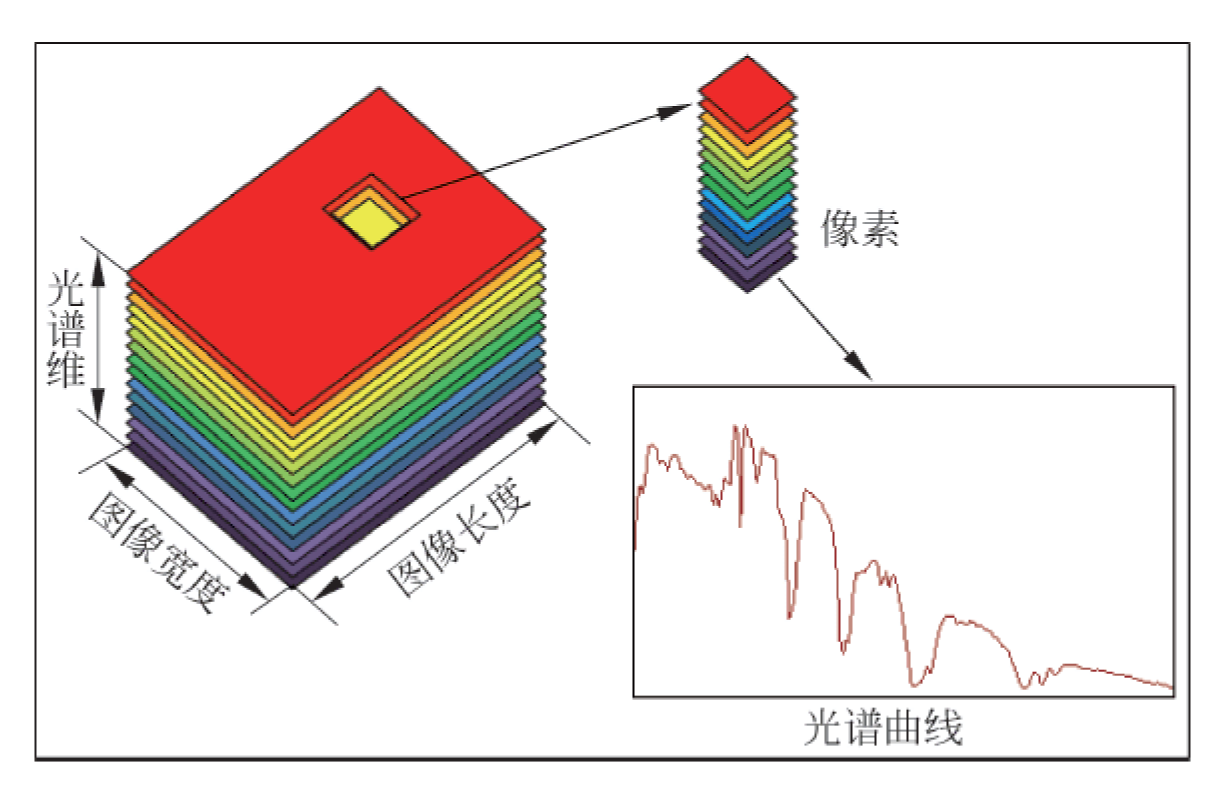


图14.1　高光谱图像三维数据的结构示意图

总之，相较于传统的遥感，高光谱遥感具有的主要特点是：

- 具有非常高的光谱分辨率，成像波段从可见光到短波红外，其波段宽度能够达到纳米的级别（10～20nm）；
- 具有光谱连续性，由于成像光谱仪得到的波段数目大，相邻的波段之间存在重叠，形成连续光谱曲线；
- 具有图谱合一的性质，同时具有丰富空间和光谱信息，使两者得以融合；
- 具有高数据维度，光谱分辨率的增加是由于波段数目的增多，也使得数据的维度越高；
- 具有波段相关性，波段之间的重叠造成波段间的光谱冗余，使波段之间存在相关性，同时，地物在空间位置上一般都是成块分布的，在成像过程中，传感器对某像素成像时会吸收该像素以及其周围像素的辐射，所以一般某像素与周围像素具有相同的类别，也就是空间相关性；
- 获得标记样本的难度大，由于标记过程需要进行实地勘察，其过程需要耗费大量的人力财力，在某些特殊情况下，比如遇到自然灾害时，获取标记基本不可能，此时，如何能够利用少量的标记样本进行分类成为一大难题。

我国在研究成像光谱仪的过程中同样屡获硕果。我国成功研制的推扫式成像光谱仪(PHI)和实用型模块化机载成像光谱仪(OMIS)为其中的典型代表。2002 年“神州”三号飞船携带中分辨率成像光谱仪(CMODIS)成功升空标志着我国已经拥有航天载光谱成像仪，在时间上仅次于美国。在我国的很多技术领域中，成像光谱仪均得到广泛的应用，同时随着我国的高光谱遥感技术的不断进步，已逐步与世界先进水平齐头并进。

14.1.2 高光谱遥感的研究目的

随着成像技术的迅速发展，高光谱图像的光谱信息和空间信息越来越丰富，具有十分重要的意义，世界各国广泛地研究和应用高光谱分辨率和高空间分辨率的高光谱图像。

在海洋应用方面，世界各国利用各种成像光谱仪，对海洋进行了同步、大面积、实时、连续而密集的海洋探测，获取海水水体质量的存在形式和变化趋势。在海洋遥感中，高光谱图像的光谱信息主要由纯水、碎屑、无机盐、浮游植物、有机物质以及矿物性悬浮体多种物质等的复杂作用产生，其中包括：各种物质的光谱信息综合作用，使得各个像素点的反射模型和光谱曲线有很大的差异，呈现出较大的随机性，各个像素点包含的物质和能量具有时变性和不确定性。人们主要利用高光谱遥感技术对海洋环境进行监测，协助海岛的测绘与管理，观测海岸线的地貌以及海岸线环境的演变。此外，利用高光谱遥感在海面浪场、潮汐、流场、风场、锋面等方面也有广泛的研究。

在资源勘探方面，人们主要利用高光谱成像仪获取高光谱遥感影像的地物光谱曲线，并以此为基础挖掘不同矿物质光谱反射曲线的差异性与相关性，建立矿物质光谱数据库，我们借助机器学习和模式识别理论，与光谱数据库中已有的不同矿物质存在的光谱曲线进行相似性匹配，实现矿物质的分类，最终快速准确地识别大面积矿物，提高了准确快速寻矿的可

能性。

在农业方面，为了快速提取作物生长信息，检测作物的长势，估算植被覆盖率和检测农作物品质，人们充分利用高光谱遥感技术，依靠光谱极高的分辨率。这样人们可以调整物资的投入量，从而达到减少浪费，增加产量，提高品质，保护环境质量和农业资源。此外，高光谱遥感技术也可以为精细农业提供农作物施肥管理，病虫害防治等应用提供科学的指导，为高科技农业的发展提供有利的技术保障和可靠的数据来源。

在军事方面，世界各国主要利用高光谱遥感技术完成战场详细侦察，识别伪装目标，探测计算目标真实温度和发射率等任务。通过高光谱遥感技术所获得所观测目标的波段是连续的，所以能通过连续波段直接精细地反映出被观测目标的光谱反射特征，从而分辨出被观测目标表面以及内部的状态，最终可以精确地识别地面目标。高光谱遥感也能够充分利用目标光谱反射特性与伪装目标光谱反射特性的不同，从而分辨出军事装备，最终成功识别出伪装目标。

总体来讲，高光谱遥感技术在人们的生产生活，社会的经济发展以及国家的国防安全等众多方面都有着重大的意义。高光谱遥感技术的发展也越来越成熟。积累了海量、丰富的高光谱遥感影像数据，这些数据广泛地应用在各个领域，所产生的作用也越来越大。因此，建立高光谱遥感技术新理论，完善高光谱遥感图像分析的方法和模型，寻找新的遥感影像处理算法，提取有效的遥感信息、对生活、经济和社会发展都具有非常重要的意义。

高光谱图像一般被视为一个三维立方体数据块，即空间域和光谱域。空间域指的是地表的物体特征，光谱域则给出了每一个观测地物的光谱曲线。因此，高光谱数据将空间域特征和光谱域信息完美地融合到了一起。然而，高光谱分类也还有很多问题要处理，其主要有：数据冗余问题、特征融合问题、标记样本数量问题、分类器选择问题。高光谱图像中有数百个光谱通道，波段之间存在着很大的相关性和冗余性，进而选择恰当的光谱通道进行分类是非常有必要的，否则大量的冗余信息会大大增加时间复杂度，影响分类速度。在高光谱数据分类任务中，我们最常用的就是对光谱信息进行分类，较少考虑图像的空间信息。因此，如何利用空间信息，将高光谱图像的空间信息和光谱信息融合在一起，提高高光谱图像的分类精度是目前面临的主要问题。分类器对于分类所需的训练样本数量要求越来越多，而实际标记的样本数量较少，无法满足训练所需的标记样本的数量，从而影响了分类器的分类性能。如何大量获取样本的标记是我们需要解决的另一问题。此外，在进行分类任务时，如何利用已有知识，选择出色的分类器来完成分类任务，并得到理想的分类精度，又是一个亟须解决的必要问题。对于高光谱图像的光谱域高维特性和信息重复性，在进行分类时，人们可以考虑主成分分析或者基于进化计算的选择方法对高光谱图像先进行降维和特征 2 学习等工作，降低高光谱图像的维度。

14.1.3 常用的高光谱数据集

本节中，我们主要介绍四个常用的公开数据集分别是 Indiana Pines 数据、Salinas 数据、PaviaU 数据和 PaviaC 数据。下面对所用实验数据进行详细介绍。

Pavia University 数据由 ROSIS 传感器获得。该数据在 2003 年的意大利帕维亚地区拍摄，原始数据的空间大小为 610×340 个像素点，光谱覆盖范围为 430～860nm，拥有 103 个波段，空间分辨率达到了 1.3m。该场景中对 9 类地物进行了标记，包括树木、土地、道路等地物。图 14.2 给出了帕维亚大学的某一波段图和其 9 类地物的真实标签图。表 14.1 给出了 9 类地物类别的详细信息。

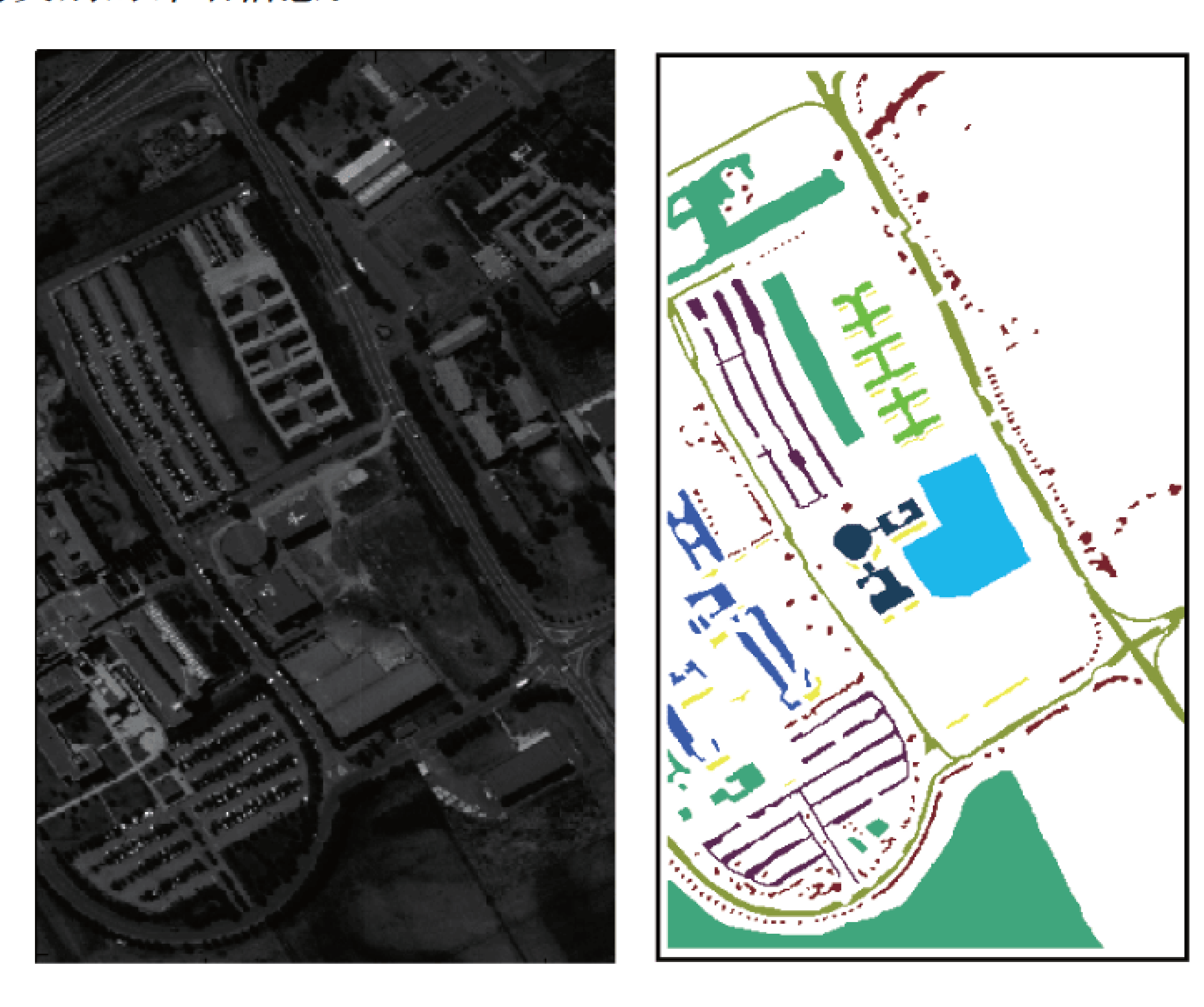


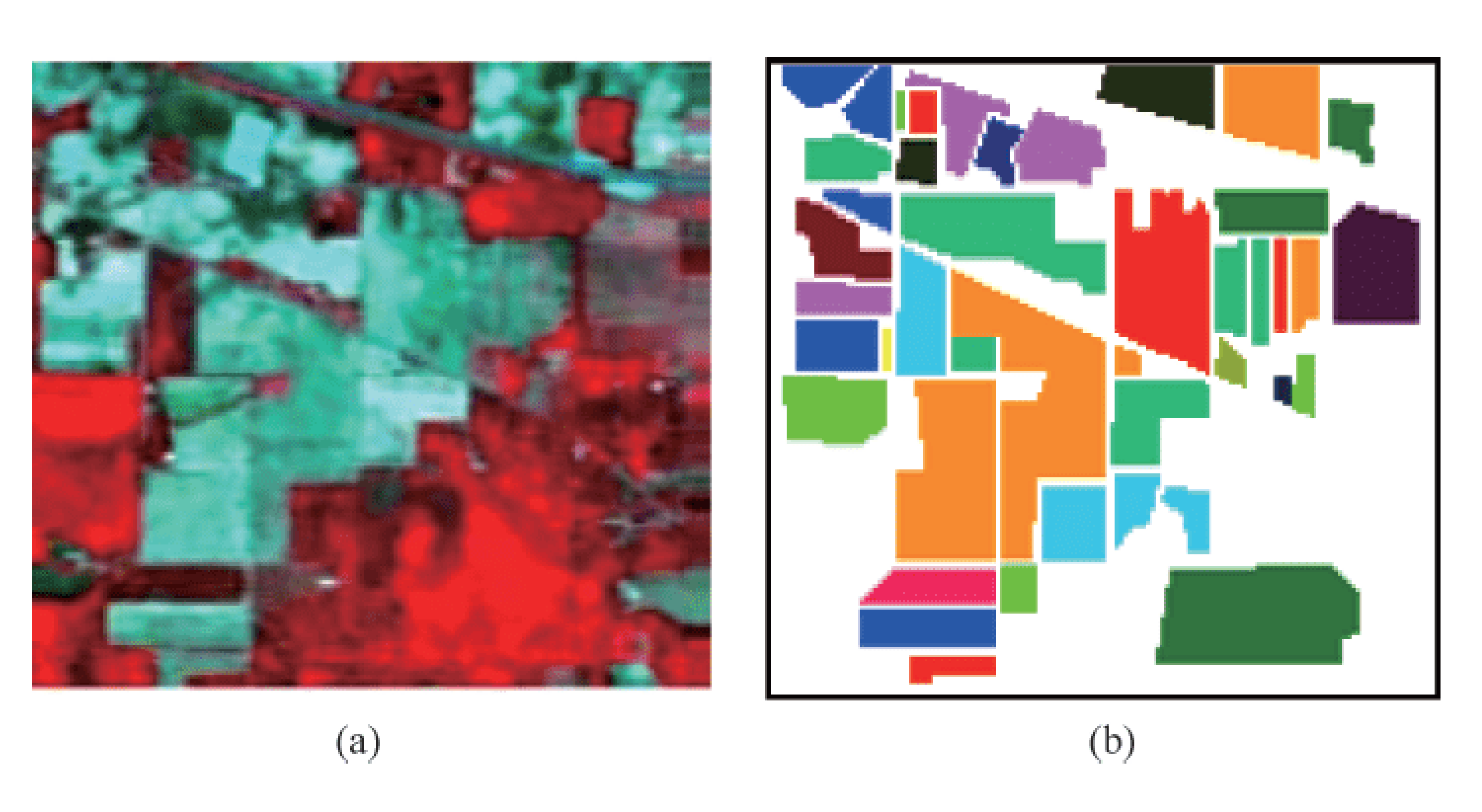
图 14.2　Pavia University 的某一波段图及其地物真实标记图

表 14.1　Pavia University 类别信息

#	类　　别	样　本　数
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare Soil	5029
7	Bitumen	1330
8	Self-Blocking Bricks	3682
9	Shadows	947

Indiana Pines 数据由 AVIRIS 光谱仪在北印第安纳州西部的 Indiana Pines 地区采集得到。图 14.3(a)给出了该数据的伪彩色图。该观测数据场景包含了森林、农田和其他自然生长的草木，还包括了两条公路和铁路，以及部分低密度的住宅，其他人工建筑和较小的道路。该数据拍摄于 6 月，因此，该场景中包含的玉米和大豆处于生长早期阶段，而且这些

作物的覆盖率小于5%。已有的真实标记图将场景中包含的地物分成了16类，并且表14.2给出了该数据详细的类别信息。而且，为了获得更好的分类效果，一些对比度较低或者噪声含量较大的波段被移除，波段数从220减少到了200。图14.3(b)给出了该数据的真实类别标记。



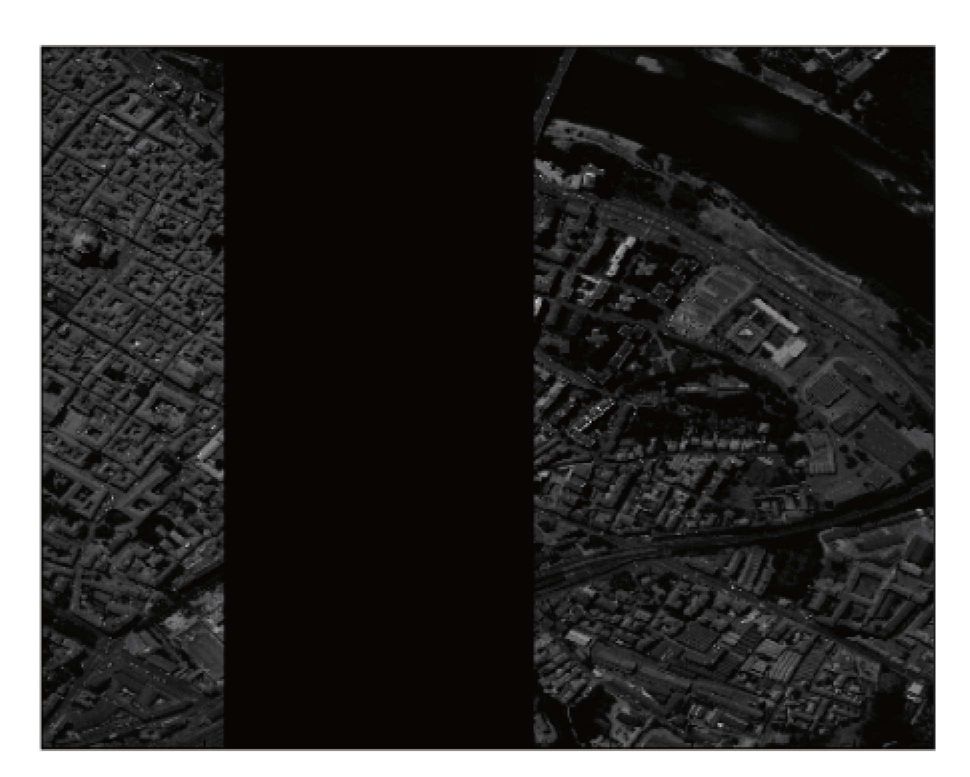
(a)　(b)

图14.3　Indiana Pines伪彩色图像及其地物的真实标记图

表14.2　Indiana Pines类别信息

#	类　别	样　本　数
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steal-Towers	93

Pavia Center数据由ROSIS传感器获得。该数据拍摄于2003年的意大利帕维亚地区，光谱覆盖范围为430～860nm，空间分辨率达到了1.3m，拥有103个波段。该场景中对9类地物进行了标记，包括树木、土地、道路等地物。图14.4给出了帕维亚的某一波段图和其9类地物的真实标签图。表14.3给出了9类地物类别的详细信息。



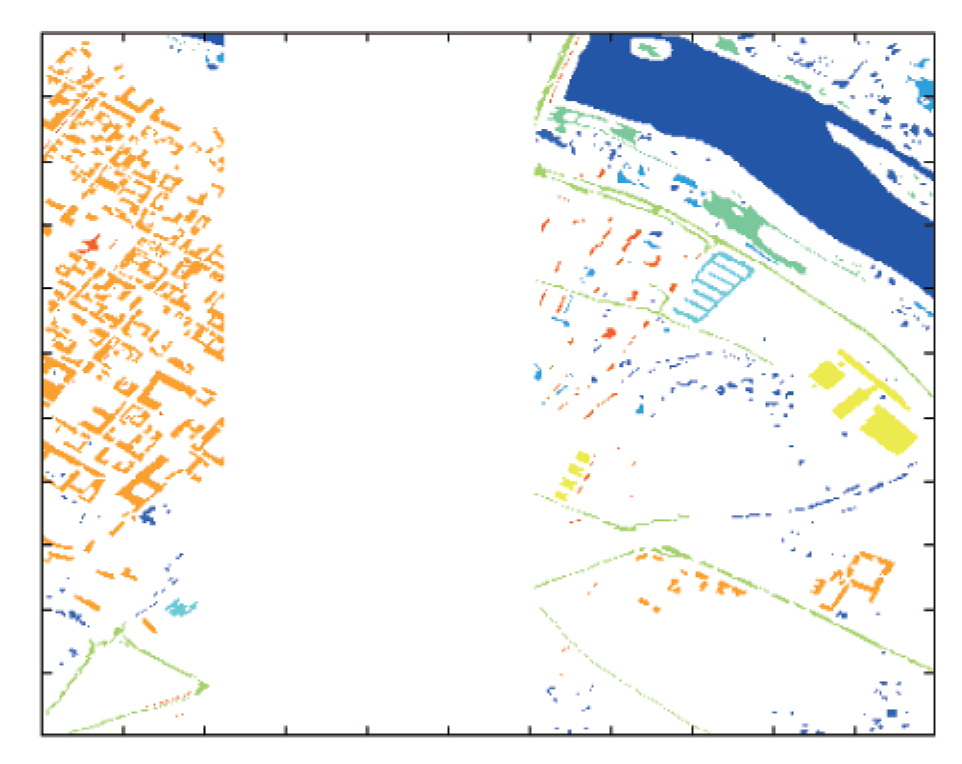


图 14.4　Pavia Center 的某一波段图及其地物真实标记图

表 14.3　Pavia Center 类别信息

#	类　别	样　本　数
1	Water	824
2	Trees	820
3	Asphalt	816
4	Self-Blocking Bricks	808
5	Bitumen	808
6	Tiles	1260
7	Shadows	476
8	Meadows	824
9	Bare Soil	820

Kennedy Space Center(KSC)数据是由 NASA 的机载可见光/红外成像光谱仪(AVIRIS)获取的，该数据于 1996 年 3 月 23 日拍摄于 Florida。原始数据的空间大小为 512×614 个像素点，覆盖面积接近 20km，空间分辨率为 18m，包含了 176 个波段，其中舍弃了大气水吸收波段，以及低信噪比波段。整个数据包含了 13 种不同的地物，包括灌木丛、沼泽、柳、水、泥潭等。表 14.4 列出了 13 种地物的详细信息。图 14.5 展示了 KSC 数据的效果图。

表 14.4　KSC 数据类别信息

#	类　别	样　本　数
1	Scrub	347
2	Willow swamp	243
3	CP hammock	256
4	Slash pine	252
5	Oak/Broadleaf	161
6	Hardwood	229
7	Swamp	105

续表

#	类　别	样　本　数
8	Graminoid marsh	390
9	Spartina marsh	520
10	Cattail marsh	404
11	Salt marsh	419
12	Mud flats	503
13	Water	927

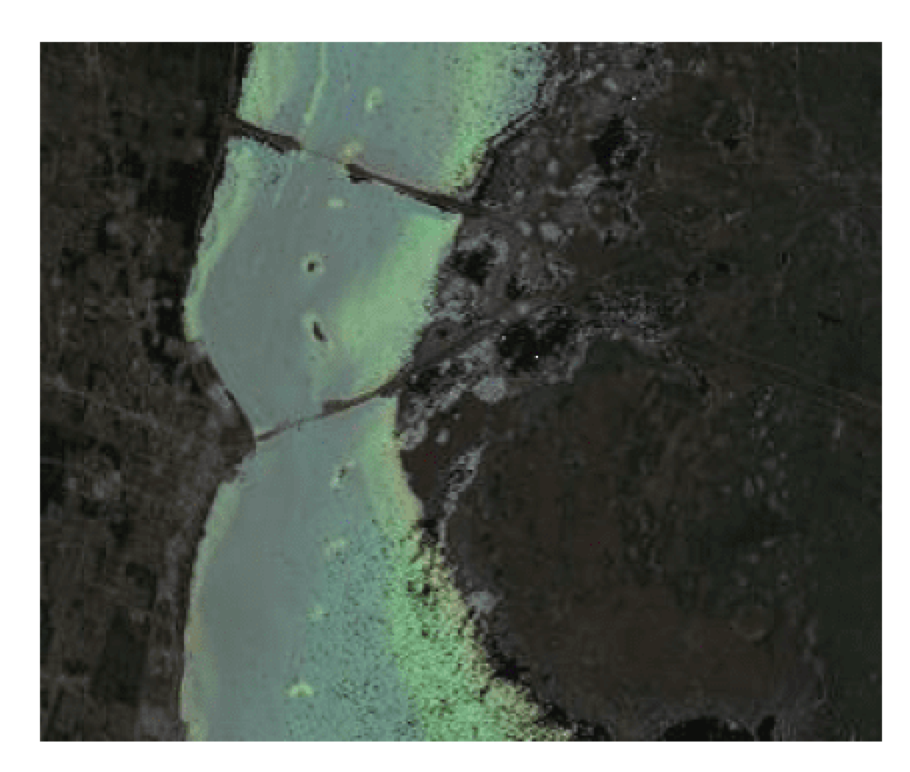



图 14.5　KSC 某一波段图及其地物真实标记图

14.2　基于深度神经网络的高光谱影像的分类

高光谱遥感技术不仅能够准确地提供所观测场景中地物的光谱性质，而且还能够反映地物之间的空间关系，实现了空和谱的结合，从而实现了高光谱影像的可靠性和丰富性。这些性质使得人们能够对地物信息进行全面的观测。因此，如何利用这些丰富的光谱信息和空间关系，对高光谱影像地物进行精确的分类是高光谱遥感技术研究领域的一大热点。高光谱图像分类技术经过几十年的研究和探索，现在已经取得了较大的发展，针对高光谱图像的特点，研究者们提出了很多的高光谱图像分类算法。总体来说，根据有无标记样本参与分类器的训练，高光谱图像分类方法大致可以分为三大类：有监督、半监督和无监督方法。

在对高光谱图像进行分类时谱间存在 Hughes 现象，也就是说，当训练样本数目有限时，分类精度随着图像波段数目的增加而增加，增加到某一个临界点后，继续增加反而会导致分类精度下降。Hughes 现象的出现通常与训练样本数目的多少和特征的维数有关。由于高光谱图像拥有更高维的谱间信息，传统的那些进行多光谱处理的方法已不再适合处理高光谱图像。现在已经有很多高光谱图像处理的方法，例如，利用传统数据降维的方法来进行特征提取，主要方法有主成分分析(Principal Components Analysis, PCA)、投影寻踪方



法(Project Pursuit,PP)、独立成分分析(Independent Component Analysis,ICA)等。最近几年一些集成学习的方法取得了不错的效果,Waske 等人使用随机特征选取的策略选取几个特征集,然后分别在特征集训练 SVM 分类器,最后通过各个分类器的结果来决策最终分类结果。上面这些方法都仅仅使用了高光谱的谱间信息。一些研究表明,高光谱的空间信息能够描述物体的空间结构属性,这些属性与谱间信息结合能够很好地提高分类精度。通常认为相邻的高光谱像元是由相似的地物构成的,因此它们的光谱特性具有很高的关联相似性。在进行高光谱图像处理的时候考虑像元间空间关系将大大提高高光谱遥感图像处理的效果。马尔可夫随机场(Markov Random Field, MRF)分类方法是一种能自动、有效利用地物空间信息的分类方法。MRF 理论是建立在空间相关性上的,图像的马尔可夫性定义像元条件概率只与其邻域像元相关,而与其他一切因素无关。基于空间相关性的 MRF 模型分类算法为遥感图像分类提供有力工具。

稀疏表示算法可以在原始的高维度空间,对高光谱数据进行分类。该算法通过使用有标记的样本组成过完备字典,通过字典中少量训练样本的线性组合表示测试样本,最后考察测试样本与其稀疏分解之间的逼近程度对测试样本进行分类。在进行稀疏表示的高光谱遥感图像分类中将像元间上下文的关系考虑进去,构建了基于上下文关系的稀疏表示模型,联合子空间追踪算法(Simultaneous Subspace Pursuit, SSP)和联合正交匹配追踪算法(Simultaneous Orthogonal Matching Pursuit,SOMP)。

现如今,深度学习在众多领域取得了很好的效果。它通过多层的神经网络来提取高层特征以描述高维数据的复杂结构。到目前为止,已经有一些深度学习的方法应用到高光谱图像分类中来,如堆栈自编码(SAE)、深度信念网(DBN)、深度卷积神经网络(DCNN)等。接下来将介绍深度学习模型在高光谱影像的分类中的应用。

14.2.1 基于堆栈自编码的高光谱影像的分类

1. 基于谱间信息的堆栈自编码模型

自编码器是深度学习领域中的一种典型算法,可以实现无监督地提取数据的特征。自编码器由一个三层前馈神经网构成,包含编码器和解码器。编码器用来对原始特征进行编码,解码器则对编码后的特征进行重构。通过解码器重构特征与原始特征之间近似的优化条件来进行无监督训练。

堆栈自编码网络是指将普通的自编码器多层连接后,再连接一个分类器所得到的神经网络。该网络拥有多层自编码器,可以实现逐层编码提取特征。

如图 14.6 所示,该模型为用于高光谱分类的堆栈自编码网络。该网络包含四个自编码器和一个逻辑回归分类器,是基于谱间信息的分类模型,输入数据为原始高光谱图像在某一像元上的光谱信息。第一层自编码器将输入像元的光谱信息映射成为第一层的特征。第一个自编码器训练结束后,只保留了编码器部分的网络。然后将编码器的输出当作下一层的输入,再次按照逐层学习的方式训练第二个自编码器,保留编码器部分的网络。如此递推,

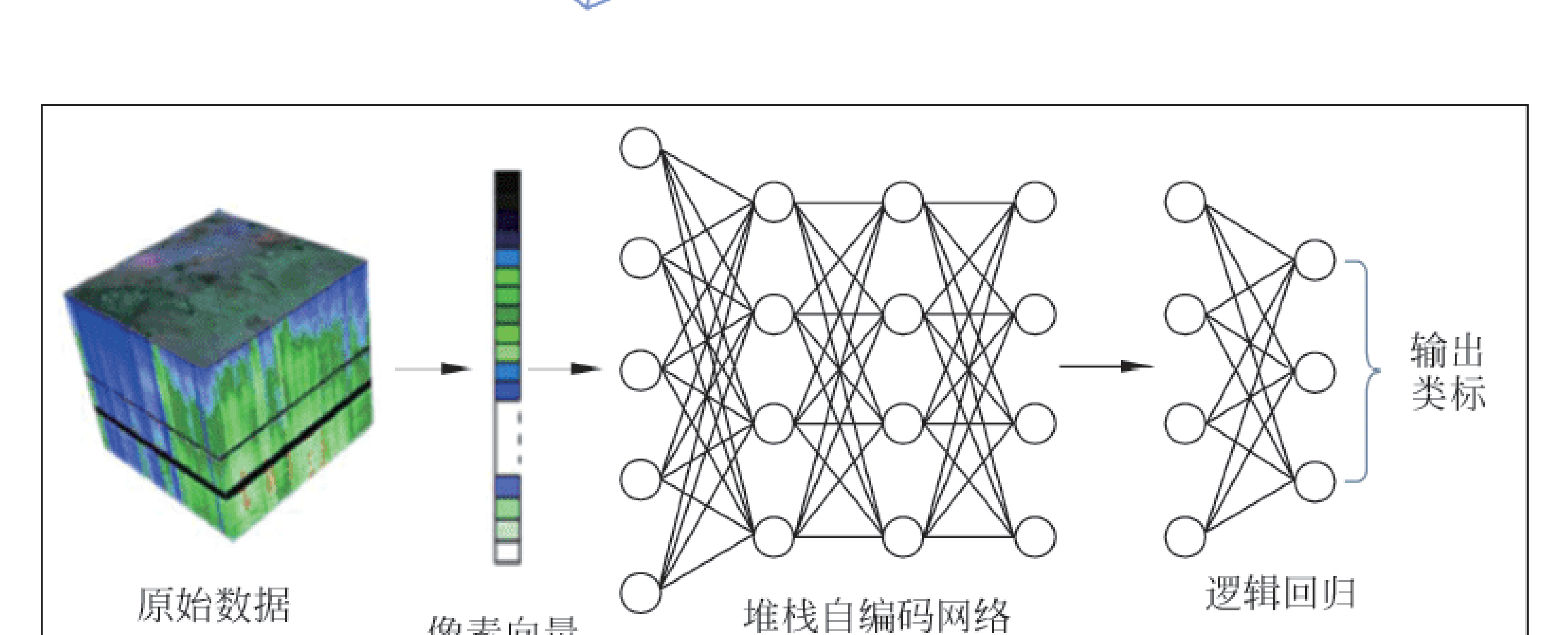


图 14.6　基于谱间信息的堆栈自编码模型架构

直至将每一层自动编码机训练完毕。像这样的训练方式可以在每一层上都得到最小化信息的损失，提取出的特征逐层地变得更抽象、具有更好的不变性。最后将提取的特征送入分类器中，得到最终分类结果。

2. 基于空间信息的堆栈自编码模型

如图 14.7 所示，该模型是基于空间信息的堆栈自编码模型，该模型主要将目标像元邻域的空间信息以图像块的形式输入到网络。由于高光谱图像谱间信息维度高，在将图像块输入到网络之前，首先沿着光谱方向对整幅图像做 PCA 变换，提取主成分。PCA 变换既能将光谱信息进行降维，又不会改变空间信息的分布。然后在压缩后的数据上目标像元为中心提取邻域图像块(图像块通常是几个到十几个像素大小)。由于该堆栈自编码器处理的是一维数据，在编码之前需要将每个像素所提取的三维数据立方体伸展成一个一维向量，以便将其输入至堆栈自编码器中。最后将由多层自编码器编码后提取的特征输入至分类器进行分类。该模型使用的是逻辑回归分类器。

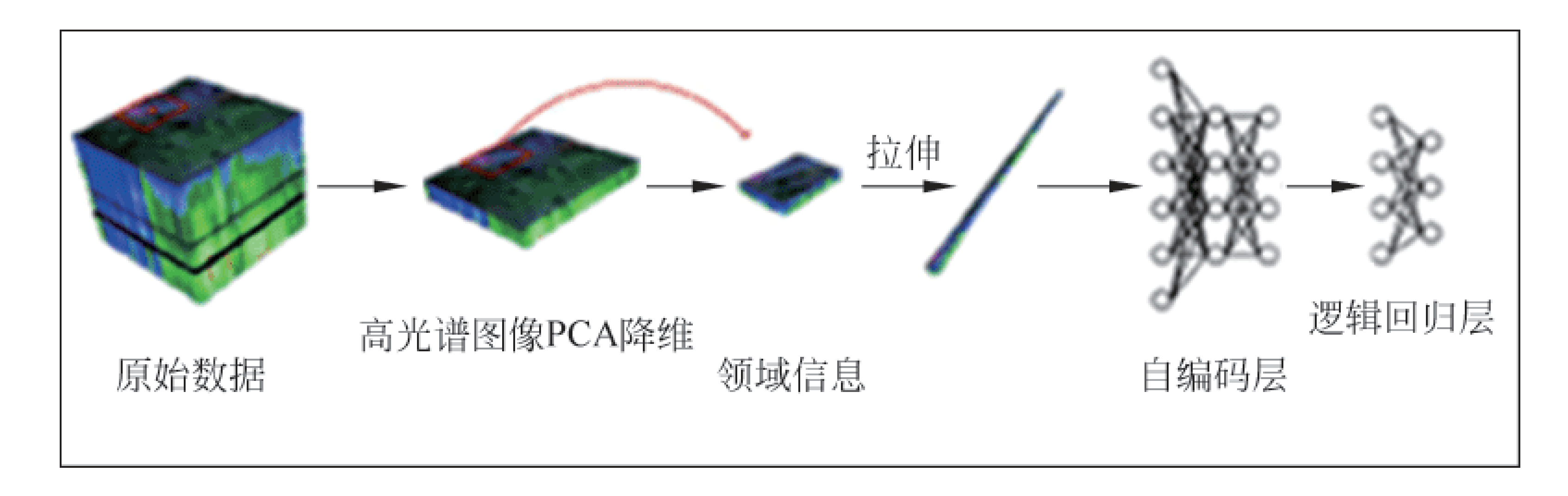


图 14.7　基于空间信息的堆栈自编码模型架构

模型的训练过程与基于谱间信息的堆栈自编码模型相似，在此不再赘述。

3. 基于空谱联合的堆栈自编码模型

图 14.8 中的两个模型分别基于谱间信息和空间信息，谱间信息能够很好地反映目标像

元的光谱属性,包含用于识别不同类型的地面的类别中最重要的信息。但是,随着技术的发展,高光谱图像的空间分辨率在逐渐提高,因此相邻像元之间的相关性也越来越大。现在很多研究表明,空间和谱间信息结合能够有力地提高高光谱图像的分类精度。

本模型将结合前面介绍的两个模型的特点,将目标像元的谱间信息与其邻域空间信息联合的形式输入到自编码网络中,以获得更加丰富的特征表示。如图 14.8 所示,模型将经过 PCA 降维后的空间特征与光谱特征连接起来,形成了一个空谱特征联合的向量。然后将联合向量作为堆栈自编码器的输入,进行整体网络训练,训练方式如上面模型所述。

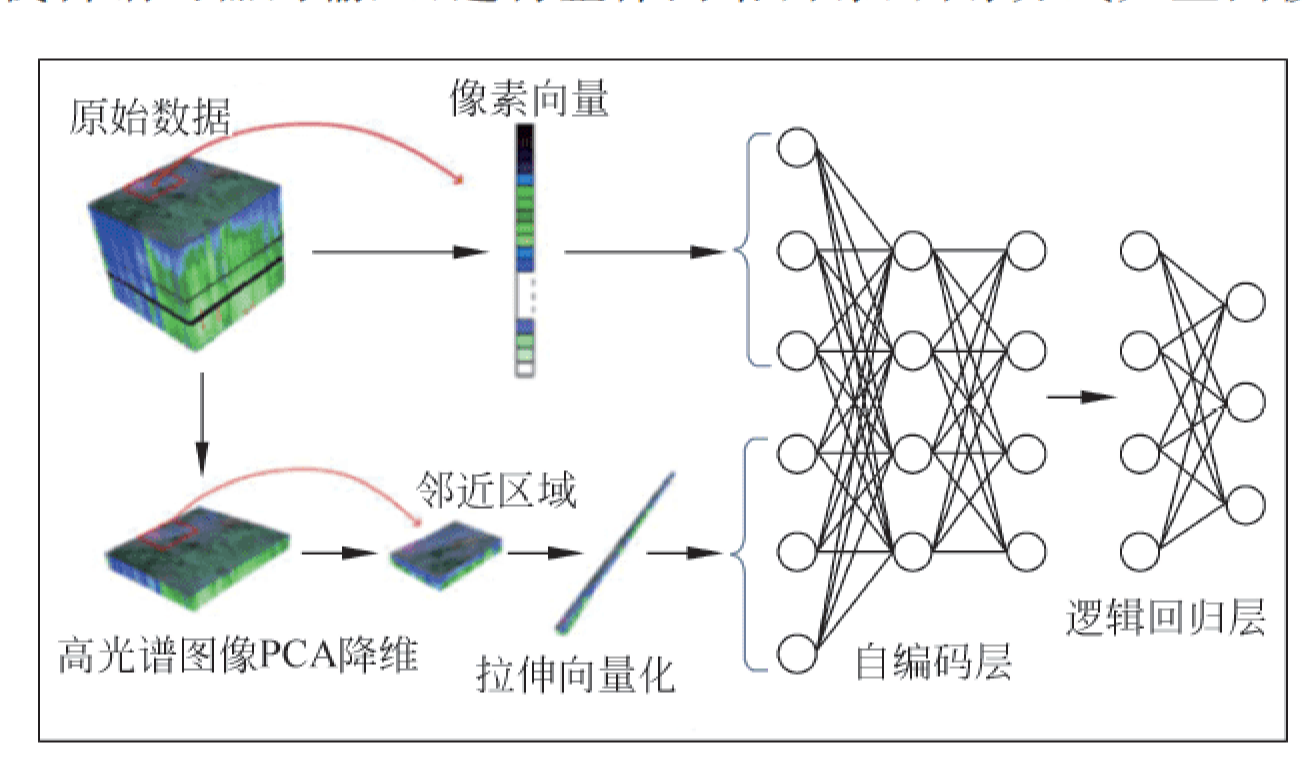


图 14.8　基于空谱联合的堆栈自编码模型架构

4. 实验结果分析

实验一:

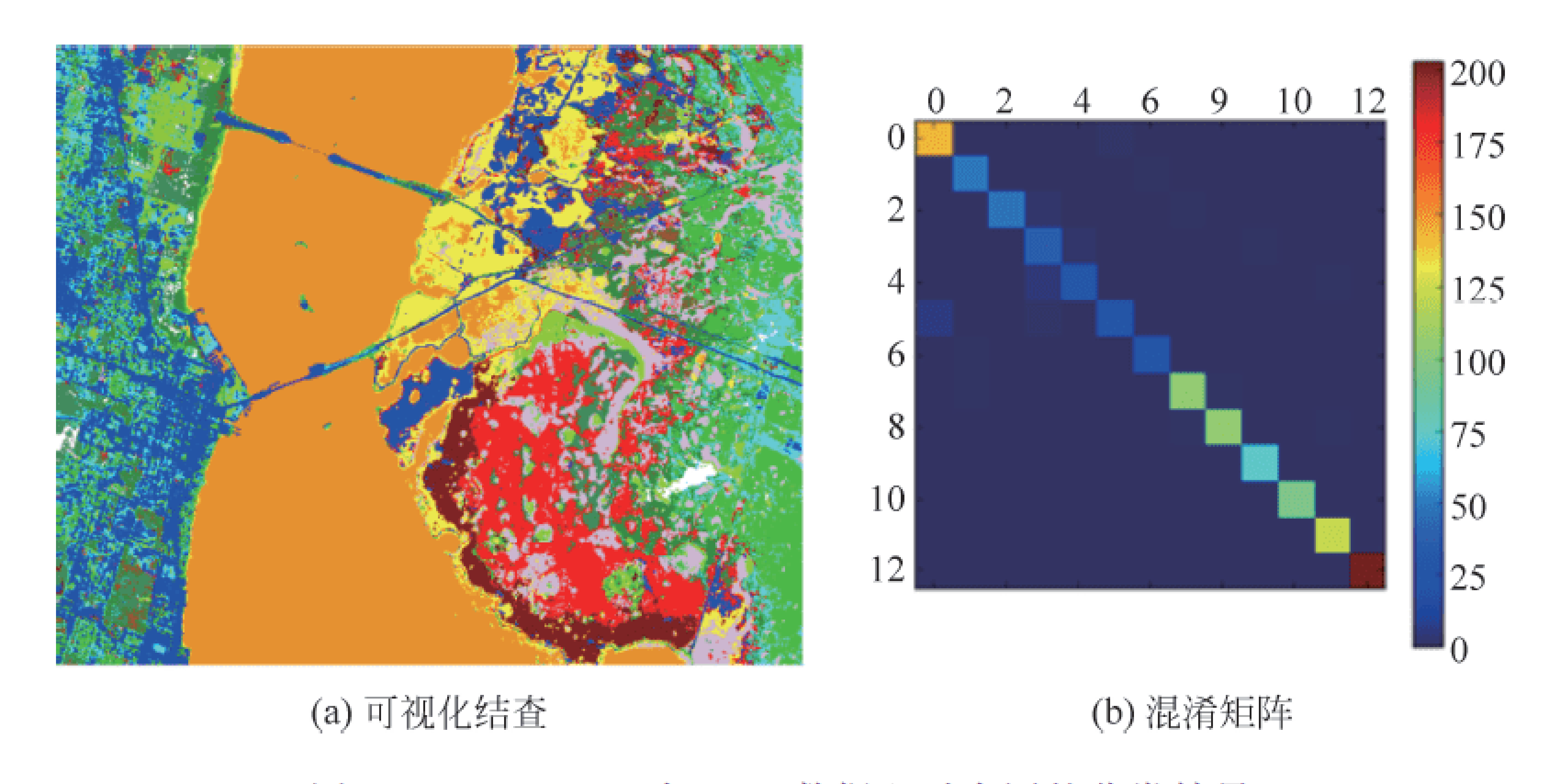
堆栈自编码网络的网络层数是影响模型性能的一个很重要因素,本实验基于谱间信息的堆栈自编码模型研究了自编码器层数对分类精度的影响。实验精度和分类所用时间如表 14.5 所示。

表 14.5　网络深度对分类精度的影响

深　度	KSC		Pavia	
	测试分类精度/%	分类所用时间/s	测试集分类精度/%	分类所用时间/s
1	94.63	0.12	92.93	0.19
2	95.45	0.15	94.95	0.27
3	96.55	0.20	94.99	0.35
4	95.27	0.22	95.16	0.42
5	93.91	0.24	95.13	0.48

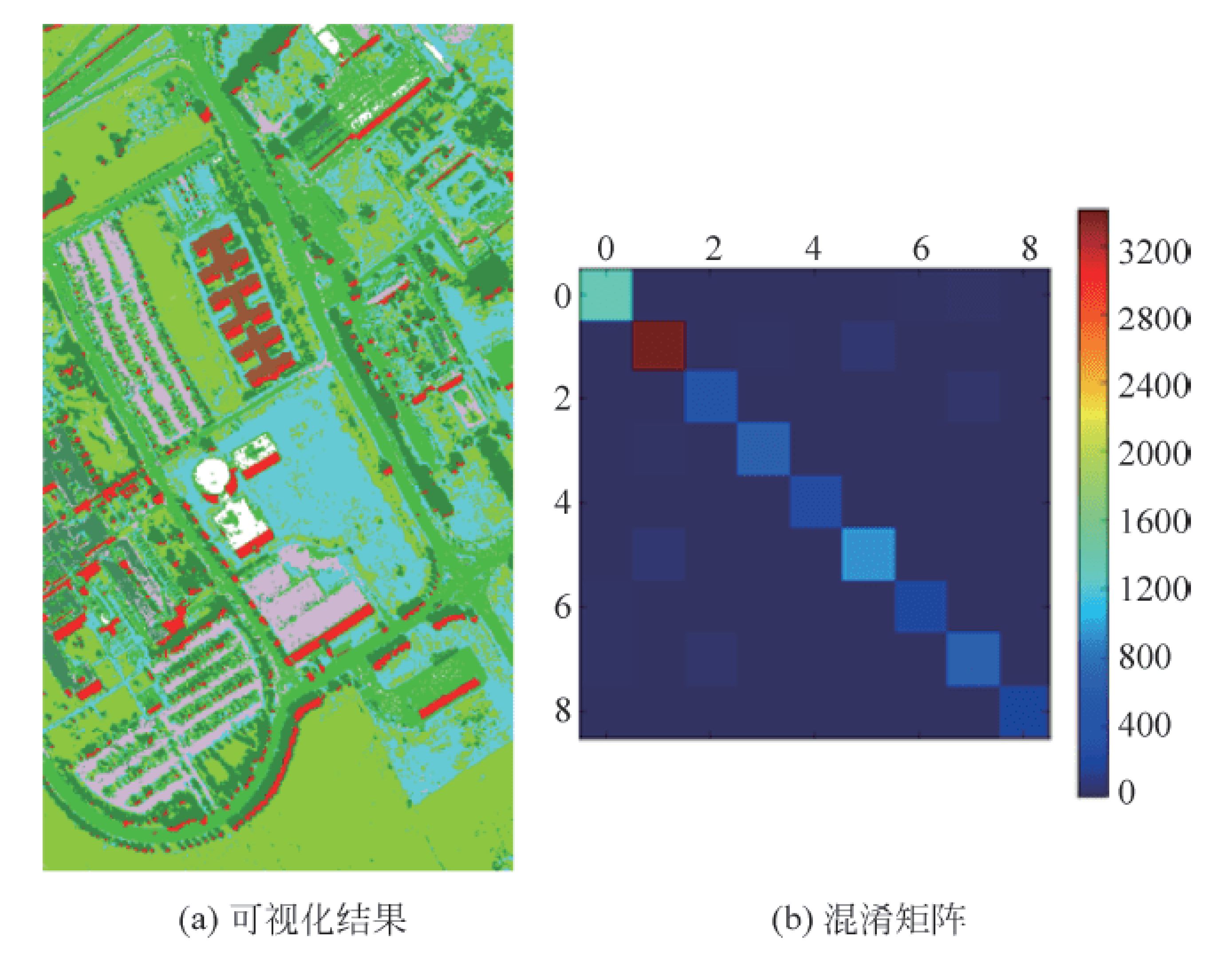
从实验结果来看,自编码器层数对分类精度的影响是很大的。对于含有 176 个波段的 KSC 数据,实验设置 20 个隐层节点,13 个输出节点。对于含有 103 个波段 Pavia 数据,实验设置 60 个隐层节点,9 个输出节点。增加网络深度也就是对网络增加隐层。表 14.5 中

结果是对实验进行了5000轮预训练和50000轮微调。KSC数据在隐层节点为3时，测试分类精度达到最高，为96.55%，此时若再增加网络层数，分类精度会有所下降。Pavia数据在隐层节点为4时，测试分类精度达到最高，为95.16%。两个数据集分类所用时间是随着网络层数的增加而逐渐增长的。SAE-LR模型对KSC数据的分析结果如图14.9所示，对Pavia数据的分析结果如图14.10所示。



(a) 可视化结查　　(b) 混淆矩阵

图14.9　SAE-LR在KSC数据上对全图的分类结果



(a) 可视化结果　　(b) 混淆矩阵

图14.10　SAE-LR在Pavia数据上对全图的分类结果

实验二：

本节通过设置对比实验SAE-LR与RBF-SVM，共执行100次重复实验。Kappa系数的统计评估箱型图绘制在图14.11(a)和图14.11(b)中。横坐标数字分别对应于：①基于

空间信息占优特征的 SAE-LR；②基于空间信息占优特征的 RBF-SVM；③基于空谱联合的 SAE-LR；④基于空谱联合的 RBF-SVM。

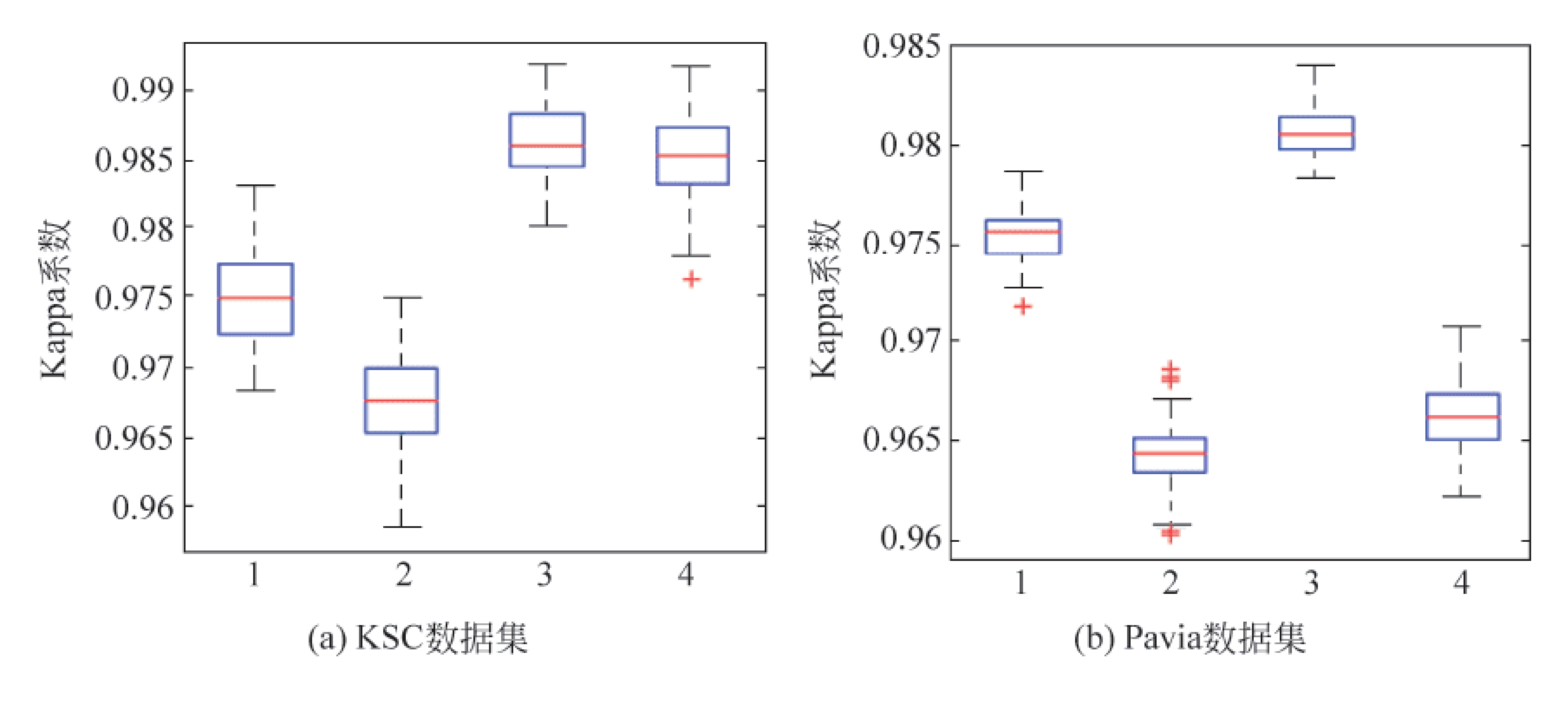


图 14.11　KSC 及 Pavia 数据集 Kappa 系数的箱形图

结果表明，在两个数据集上，空谱联合分类的精度总是高于空间信息占优的分类，并且能够在置信区间内稳定地达到较高精度。而基于空间信息占优特征的 RBF-SVM 在两个数据集上得到的分类精度最低。再对比两个分类方法 SAE-LR 与传统的 SVM，可以看出 SAE-LR 总是具有更高的精度，这也体现了堆栈自编码的优势。

接下来的实验主要对比了基于纯光谱的分类算法、基于空间信息的分类算法和空谱联合的分类算法，这三个模型分别在两个通用数据集上的分类结果对比，主要评价指标有：总体精度(OA)、平均精度(AA)、Kappa 系数以及在每一类上的分类精度。

在表 14.6 和表 14.7 中我们可以看到，无论是对于 KSC 数据还是 Pavia 数据，基于空谱信息联合的堆栈自编码网络模型在总体精度(OA)、平均精度(AA)及 Kappa 系数上都获得了最好的结果。对于 KSC 数据来说，基于空谱信息联合的 SAE-LR 比 RBF-SVM 能获得更好的分类结果，三个指标分别高出 0.15%、0.50%、0.16%。同时，对于一些分类结果不好

表 14.6　SAE-LR 及 SVM 用于三种信息的分类结果(在 KSC 数据上)

评价指标	SAE-LR			RBF-SVM		
	谱间信息	空间信息	空谱联合	谱间信息	空间信息	空谱联合
AA	0.9718	0.9782	0.9924	0.9673	0.9709	0.9909
OA	0.9462	0.9654	0.9868	0.9401	0.9571	0.9818
Kappa	0.9685	0.9756	0.9915	0.9635	0.9675	0.9899
C1	0.9862	0.9931	1.0000	0.9744	0.9862	0.9931
C2	0.9803	0.9804	0.9804	0.9608	0.9412	1.0000
C3	0.9245	0.9434	0.9623	0.9057	0.8868	0.9811
C4	0.9047	0.9286	0.9524	0.8679	0.9762	0.9286
C5	0.8333	0.8889	0.9444	0.5882	0.8056	0.8611

续表

评价指标	SAE-LR			RBF-SVM		
	谱间信息	空间信息	空谱联合	谱间信息	空间信息	空谱联合
C6	0.7500	0.8889	1.0000	0.9362	0.9722	1.0000
C7	0.9583	1.0000	1.0000	1.0000	1.0000	1.0000
C8	0.9814	0.9352	0.9889	0.9888	0.9908	1.0000
C9	0.9816	1.0000	1.0000	1.0000	1.0000	1.0000
C10	1.0000	1.0000	1.0000	1.0000	0.9730	1.0000
C11	1.0000	1.0000	1.0000	1.0000	0.9271	1.0000
C12	1.0000	0.9919	1.0000	1.0000	0.9837	1.0000
C13	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

表 14.7　SAE-LR 及 SVM 用于三种信息的分类结果(在 Pavia 数据上)

评价指标	SAE-LR			RBF-SVM		
	谱信息	空间信息	空谱联合	谱信息	空间信息	空谱联合
AA	0.9527	0.9805	0.9863	0.9531	0.9798	0.9858
OA	0.9427	0.9728	0.9805	0.9370	0.9737	0.9750
Kappa	0.9386	0.9746	0.9821	0.9389	0.9737	0.9815
C1	0.9531	0.9830	0.9753	0.9619	0.9823	0.9879
C2	0.9768	0.9948	0.9966	0.9821	0.9905	0.9948
C3	0.8578	0.9194	0.9573	0.8047	0.9526	0.9265
C4	0.9706	0.9883	0.9894	0.9554	0.9824	0.9956
C5	1.0000	1.0000	1.0000	0.9928	1.0000	1.0000
C6	0.9315	0.9648	0.9863	0.9207	0.9569	0.9883
C7	0.9263	0.9535	0.9457	0.9041	0.9380	0.9380
C8	0.8922	0.9561	0.9787	0.9216	0.9747	0.9774
C9	0.9759	0.9952	0.9952	0.9902	0.9856	0.9663

的类别，基于空谱信息联合的模型能够有很大的提升，如 KSC 数据的 C6 在基于谱信息和空间占优信息模型上的分类结果分别为 75%和 88.89%，而在基于空谱信息联合的堆栈自编码网络模型上能达到 100%的准确率。

图 14.12 给出了上述三个模型在 KSC 与 Pavia 两幅图像上分类结果的可视化结果。由左到右依次表示基于谱间信息的堆栈自编码模型、基于空间信息的堆栈自编码模型、基于空谱联合的堆栈自编码模型分类结果，各个模型分别选择最好堆栈自编码网络参数。从分类结果图中可以看出，基于空谱联合的堆栈自编码模型能够很好地使用数据的空间信息和光谱信息，对于分散的嘈杂点和影子都能够很好地处理。在此需要注意的是：空间邻域的大小对分类结果影响很大，邻域过大虽然能够很好地去除杂点，但是会损失掉物体的形状和细节。邻域过小，则去除杂点的能力会有所损失。因此，空间邻域的大小需要根据实际高光

谱图像有所调整。

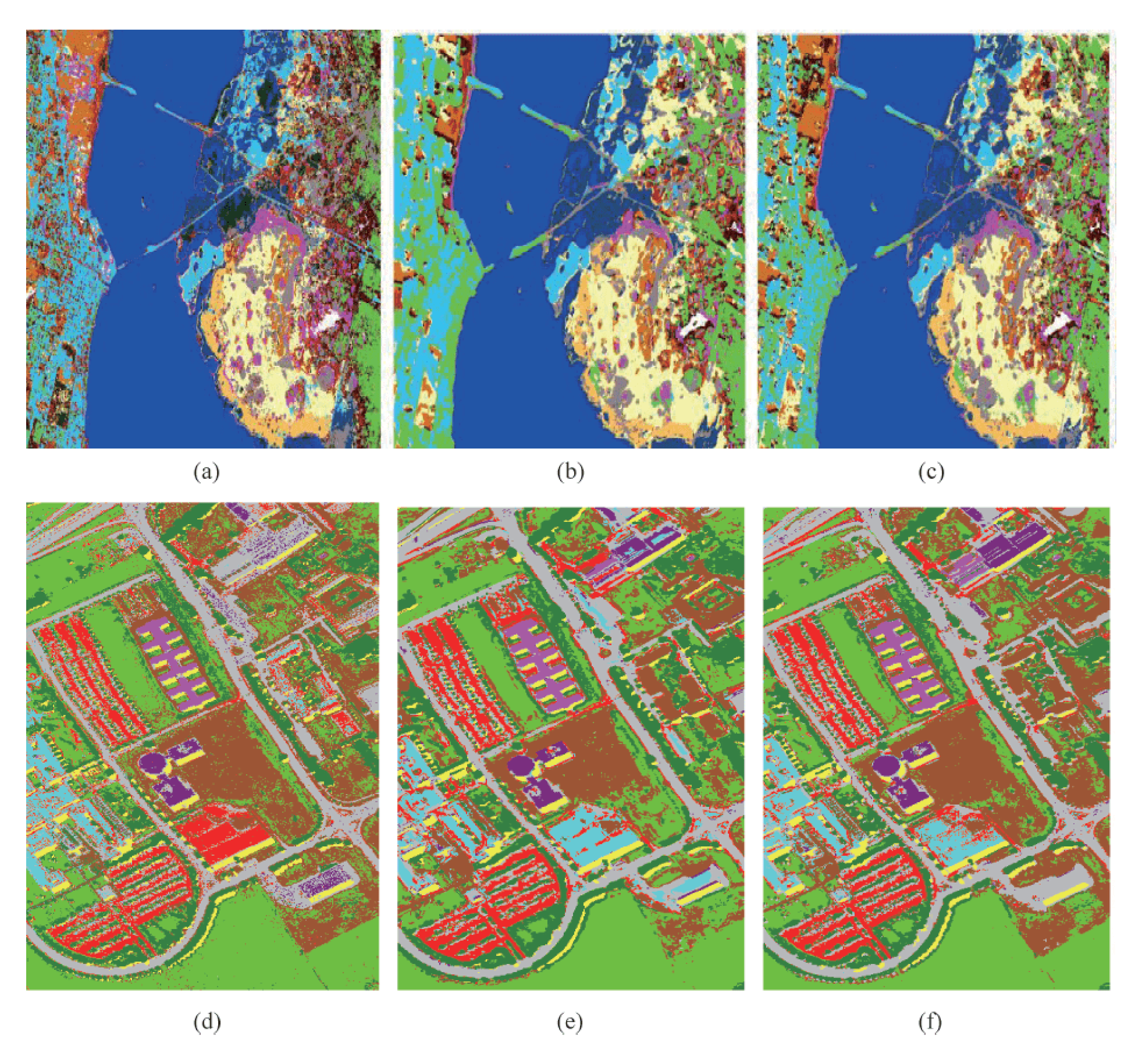


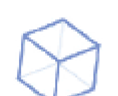
图 14.12　基于光谱信息、空间信息以及空谱联合的分类结果图

14.2.2　基于卷积神经网络的高光谱影像的分类

卷积神经网络已经在自然图像处理领域取得了很好的效果，它可以说是当下使用最多的深度学习模型。其局部感受野和权值共享策略，减少了大量的网络模型参数。陈雨时等人提出使用三维卷积神经网络提取高光谱图像的空谱特征，并取得了很好的分类结果。本节将按照陈雨时论文的思路介绍一下卷积神经网络在高光谱图像分类中的应用。

1. 基于谱间信息的一维卷积神经网络模型

通常情况下，一个深度神经网络包含多个卷积流，一个卷积流包含卷积和池化两种操作。接下来我们首先介绍一下一维卷积操作。以下公式表示计算第 i 层第 j 个特征图在第



x 个位置上的值 $V_{i,j}^{x}$：

$$V_{i,j}^{x}=g\left(b_{i,j}+\sum_{m}\sum_{p}^{P_i-1}W_{i,j,m}^{p}\cdot V_{i-1,m}^{x+p}\right) \tag{14.1}$$

$$g(x)=\tanh(x)=\frac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}} \tag{14.2}$$

其中 m 表示连接当前神经元的上一层的特征图，$W_{i,j,m}^{p}$表示 p 与第 m 个特征图的连接权值，P_i 表示的是谱间维度的卷积核的长度，$b_{i,j}$代表第 i 层第 j 个特征图的偏置。池化操作能够减少特征图的特征维数，每一个池化操作都与上一层的卷积操作有关系，最常见的池化操作是最大化池化，计算方式如下：

$$a_j=\max_{N\cdot 1}(a_i^{n\cdot 1}\cdot u(n,1)) \tag{14.3}$$

其中 $u(n,1)$代表处理卷积层的窗函数，a_j 代表邻域信息内的最大值。

基于谱间信息的一维卷积神经网络模型如图 14.13 所示，以高光谱图像像素点的谱间向量作为网络输入，以类别的数目作为模型的输出。该模型包含多个卷积流和一个逻辑回归分类器。如图 14.13 所示，网络包含两个卷积和两个池化过程。第一层卷积包含 3 个卷积核，第二层卷积包含 6 个卷积核。

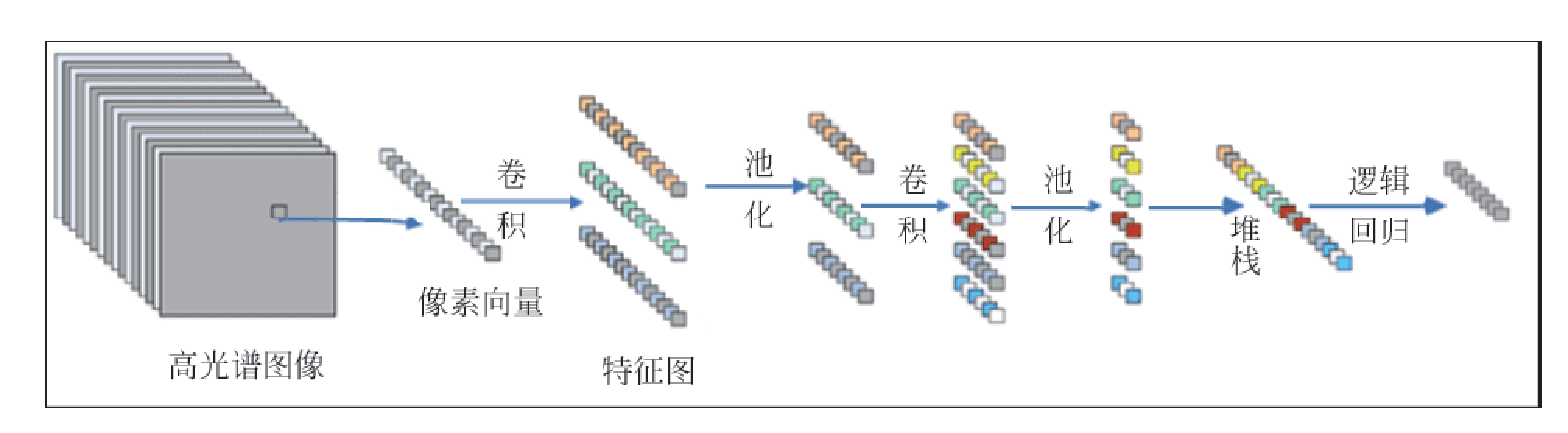


图 14.13 基于谱间信息的一维卷积神经网络模型架构

本模型只考虑了高光谱图像的谱间信息，经过几层的卷积和池化之后，输入向量最终被转化为一个包含谱间特征的向量，最后经过分类器的处理得到分类结果。该模型使用 BP 算法进行训练时的参数优化。在一维的卷积神经网络模型中，使用了局部连接和权值共享的策略来提高模型的鲁棒性。

2. 基于空间信息的二维卷积神经网络模型

与一维卷积网络形似，二维卷积流同样包含卷积和池化两个操作。当计算第 i 层第 j 个特征图在(x,y)位置上的值 $V_{i,j}^{xy}$时，计算公式如下：

$$V_{i,j}^{xy}=g\left(b_{i,j}+\sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}W_{i,j,m}^{p,q}\cdot V_{i-1,m}^{(x+p)\cdot(y+q)}\right) \tag{14.4}$$

其中 m 表示第 $i-1$ 层上与当前特征图相连的特征图，$W_{i,j,m}^{p,q}$代表与(p,q)位置相连的第 m 个特征图的连接权值，P_i 和 Q_i 代表卷积核的长度和宽度，$b_{i,j}$代表第 i 层第 j 个特征图的



偏置。

如图 14.14 所示是基于空间信息的二维卷积神经网络模型的网络结构图，该模型是针对一个通道的谱信息进行卷积操作的。首先对原始的高光谱图像数据在谱方向上进行主成分分析，提取主成分。然后以目标像元为中心选取图像块，并将图像块的第一个主成分作为网络输入进行卷积操作，经过两层的二维卷积和池化操作之后输入图像块被转化为一个特征向量，最后将得到的特征向量输入到逻辑回归分类器，得到分类结果。通常情况下，卷积核的尺寸为 4×4 或者 5×5，池化的核的尺寸为 2×2。

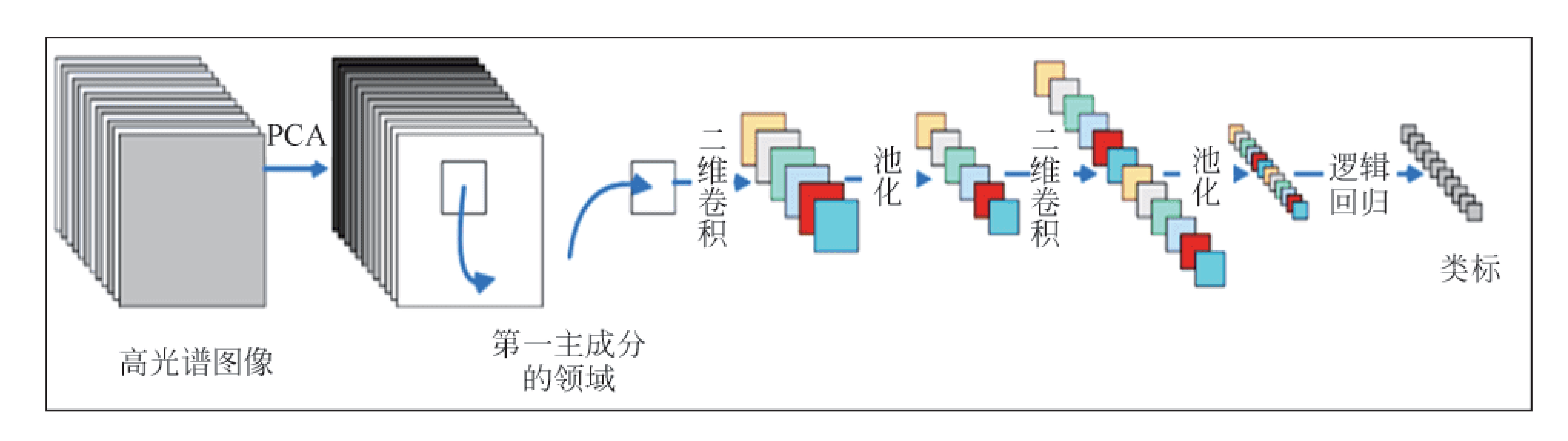


图 14.14　基于空间信息的二维卷积神经网络模型架构

3. 基于空谱联合的三维卷积神经网络模型

从上面两个卷积神经网络模型可以看出，一维卷积神经网络关注于提取谱间特征，二维卷积神经网络关注于提取像素点局部空间特征。高光谱图像包含空间信息和谱间信息，在这一部分，我们将介绍同时关注于提取空间和谱间信息的三维卷积神经网络。

图 14.15 表示了二维卷积和三维卷积操作的区别，图 14.15(a)表示一个二维卷积过程，该模型是将输入图像的一个通道经过一个卷积核操作之后映射到另一个特征图。图 14.15(b)表示一个三维卷积过程，输入图像拥有三个通道的谱间信息(Band1，Band2，Band3)，经过这三个通道的数据分别经过两个卷积核操作之后可以得到两个特征图。在进行三维卷积操作时，由于输入数据为三维数据，当计算神经网络第 i 层第 j 个特征图在(x,y,z)点值 $V_{i,j}^{xyz}$ 的计算公式如下：

$$V_{i,j}^{xyz} = g\left(b_{i,j} + \sum_{m}\sum_{p=0}^{P_i-1}\sum_{q=0}^{Q_i-1}\sum_{r=0}^{R_i-1} W_{i,j,m}^{p,q,r} \cdot V_{i-1,m}^{(x+p),(y+q),(z+r)}\right) \tag{14.5}$$

其中 m 表示第 $i-1$ 层上与当前特征图相连的特征图，P_i 和 Q_i 代表卷积核在空间上的长度和宽度，R_i 代表卷积核在光谱维度上的尺寸，$W_{i,j,m}^{p,q,r}$ 代表与(p,q,r)相连的第 m 个特征图的连接权值，$b_{i,j}$ 代表第 i 层第 j 个特征图的偏置。

如图 14.16 所示是基于空谱联合的三维卷积神经网络模型的网络结构图。该模型首先以目标像元为中心选取 $K\times K\times B$ 的邻域块作为网络的输入，K 表示图像块在空域的尺寸，B 表示高光谱图像光谱维度。然后经过两层的卷积和池化操作之后将结果拉成一列得到特征向量，最后将得到的特征向量输入到逻辑回归分类器，得到分类结果。

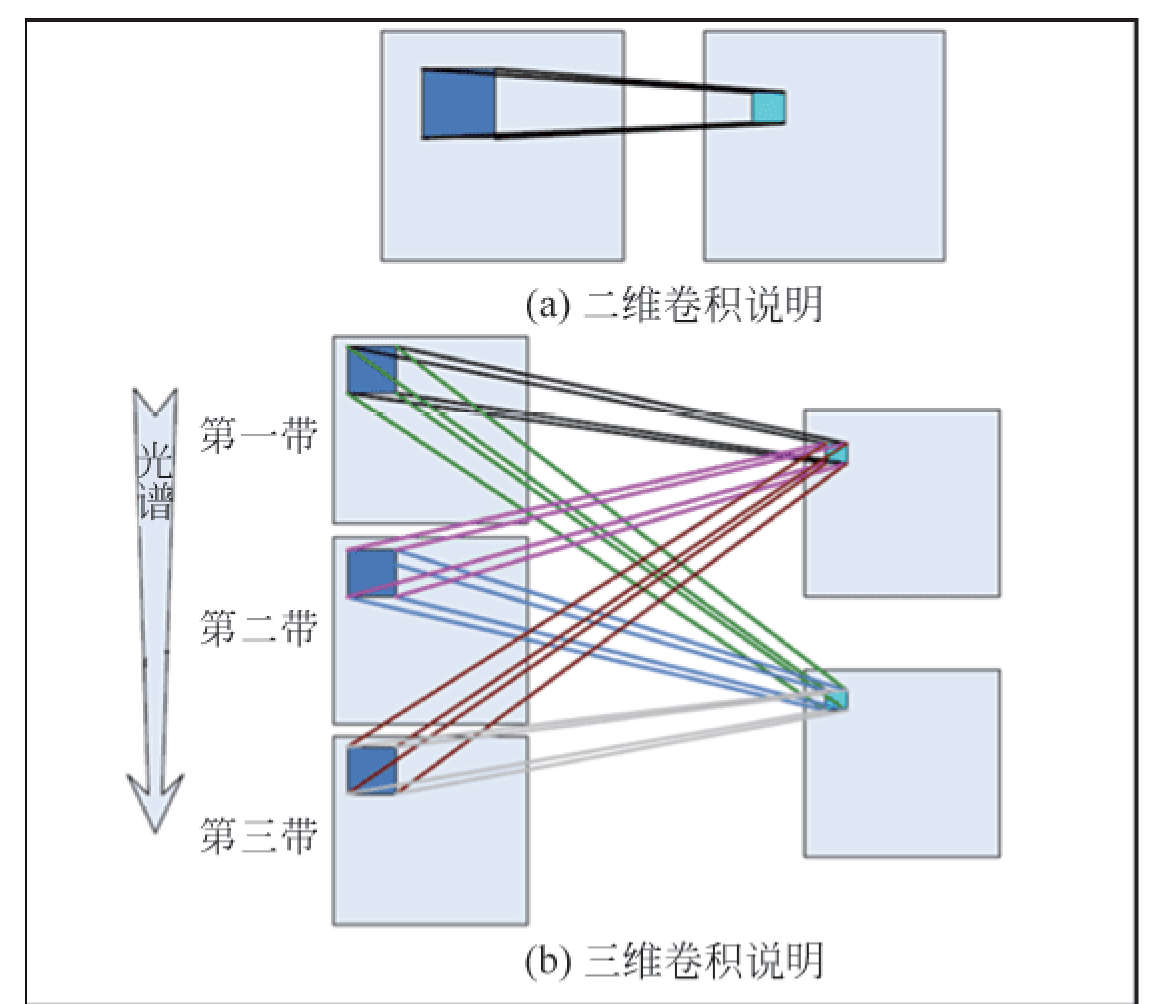


图 14.15　二维卷积和三维卷积操作示意图

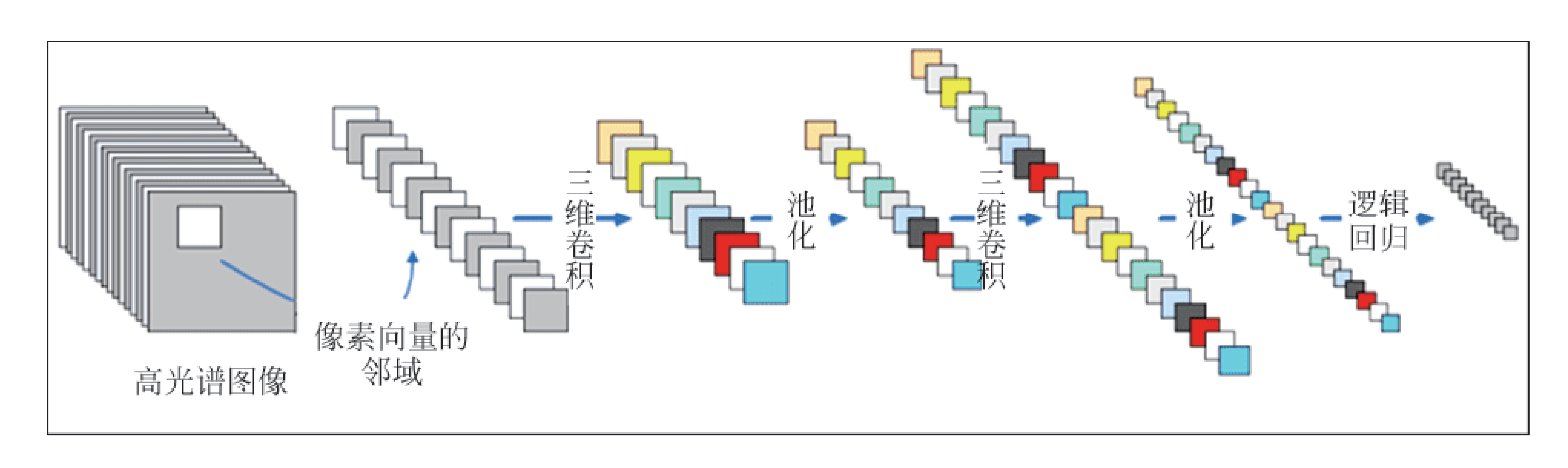


图 14.16　基于空谱联合的三维卷积神经网络模型架构

4. 实验设计及结果分析

实验一：卷积神经网络的层数是影响模型性能的一个很重要因素，本实验基于谱间信息的一维卷积网络模型研究了卷积网络层数对分类精度的影响。实验在三个通用高光谱数据上进行测试，对比了卷积层数为{1,2,3,4,5,6}的实验效果。实验精度和网络层数如图 14.17 所示。

由实验结果可知，Indian Pines 适合的卷积层数为 4，学习率设置为 0.005，迭代训练 700 次。Pavia University 最适合的卷积层数为 3，学习率为 0.01，迭代次数为 300。当卷积层数为 3 时，KSC 数据达到最高准确率，该实验学习率为 0.001，迭代次数为 600。

表 14.8 表示针对三个数据集的网络结构表，以 Indian Pines 为例，其网络结构为：I1→C2→S3→C4→S5→C6→S7→C8→S9→C10→S11→F12→O13。其中 I1 表示输入层，C 表示卷积层，S 表示池化层，F12 表示全连接层，O13 表示整个网络的输出层。

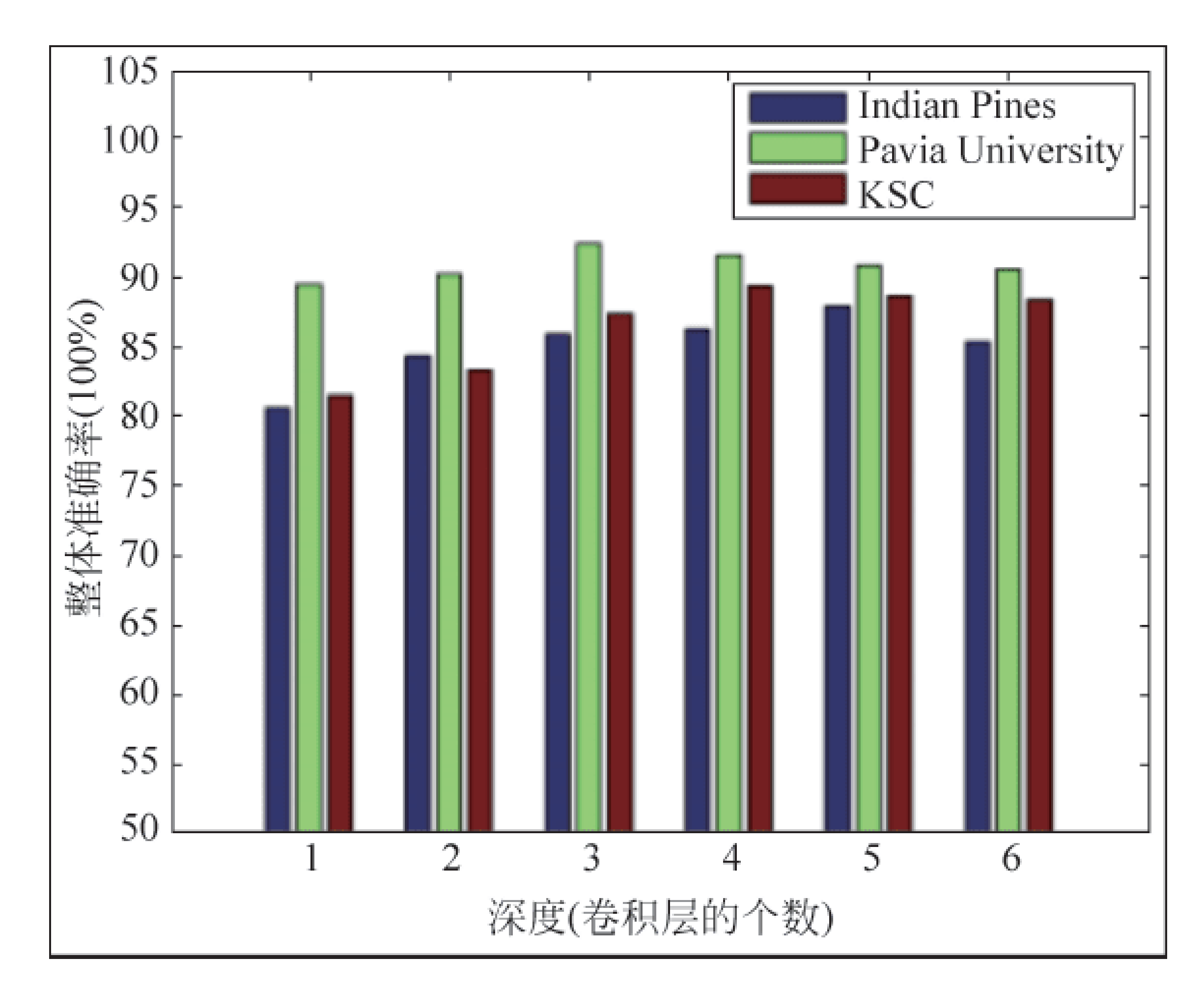


图 14.17　网络深度对分类精度的影响

表 14.8　一维卷积神经网络的网络结构参数表

Layer Name		I1	C2 S3	C4 S5	C6 S7	C8 S9	C10 S11	F12	O13
Kernel Size	Indian Pines	1 * 200	1 * 5 1 * 2	1 * 5 1 * 2	1 * 4 1 * 2	1 * 5 1 * 2	1 * 4 1 * 1	Fully connected	1 * 16
	Pavia University	1 * 103	1 * 8 1 * 2	1 * 7 1 * 2	1 * 8 1 * 2	—	—	Fully connected	1 * 9
	KSC	1 * 176	1 * 9 1 * 2	1 * 9 1 * 2	1 * 9 1 * 2	1 * 10 1 * 2	—	Fully connected	1 * 13
Feature Map		1	6	12	24	48	96	256	1

实验二：本实验是基于空间信息的二维卷积神经网络模型，对比了 RBF-SVM、EMP-RBF-SVM、2D-CNN-LR 三个模型在 Indian Pines、Pavia University、KSC 三个实验数据上的实验效果。各层网络结构参数如表 14.9 所示。

表 14.9　二维卷积神经网络卷积流参数表

No.	Convolution	ReLU	Pooling	Dropout
1	4 * 4 * 32	Yes	2 * 2	No
2	5 * 5 * 64	Yes	2 * 2	50%
3	4 * 4 * 128	Yes	No	50%

在实验中，我们选取以目标像元为中心，大小尺寸为(27×27)的图像块作为网络的输入，输入图像被归一化到[−0.5,0.5]，学习率为0.01，训练的迭代次数为200。由于输入图像块尺寸较小，我们使用了三个二维卷积层和两个池化层，经过卷积和池化之后，我们将提取的特征转化为一个128维的特征向量用于分类器处理。

实验随机选取20组训练数据和测试数据，并在三个模型上进行测试，实验结果以"平均数±方差"的形式给出。OA、AA和Kappa结果如表14.10所示。

表14.10　二维卷积神经网络在三个数据集上的分类结果

数　据　集	模　　型	RBF-SVM	EMP-RBF-SVM	2D-CNN-LR
Indian Pines	OA/%	81.14±1.47	84.47±0.99	89.99±1.62
	AA/%	85.28±1.84	87.17±1.29	97.19±0.38
	K×100	83.73±1.23	86.72±1.07	87.95±1.90
	Runtime/min	2.77	3.22	5.95
Pavia University	OA/%	91.34±0.61	93.03±0.47	94.04±0.69
	AA/%	93.87±0.87	96.88±0.79	97.52±0.25
	K×100	89.83±1.9	90.28±1.23	92.43±0.86
	Runtime/min	6.57	8.27	14.12
KSC	OA/%	88.09±0.58	93.15±0.85	94.11±0.90
	AA/%	82.51±1.07	90.51±1.11	91.98±1.34
	K×100	86.74±0.64	85.36±1.73	93.44±1.00
	Runtime/min	1.21	1.69	3.01

从上面的实验结果可以看出，2D-CNN-LR相比于其他两种RBF-SVM分类结果有明显提高，OA、AA和Kappa均优于对比实验。但是，二维卷积的实验时间长于其他两个对比模型。

实验三：本实验是基于空谱联合的三维卷积神经网络模型，该模型以目标像元为中心提取图像块，获得的图像块既包含目标像元的空间邻域信息，又涵盖了目标像元的谱间信息，经过网络模型处理之后提取空谱联合特征，再送入分类器来判别分类类别。

在实验进行之前，我们首先将高光谱原始数据归一化到[−0.5,0.5]之间，然后以目标像元为中心选取图像块，块的维度与原始数据维度相同，如Indian Pines数据我们可以选取图像块的尺寸为(27×27×200)，学习率设为0.003，训练迭代次数为400次。其他参数如表14.11所示。

输入模型的图像块的尺寸是决定分类结果的一个很重要的因素，它决定目标像元邻域信息的多少。在其他参数固定的前提下，我们设计了如下实验，用来验证图像块尺寸对分类结果的影响。我们用图像块半径作为变量来设计该实验。如图14.18所示，分别在三个数据集上以半径为$W=\{11,12,13,14,15\}$进行实验分析。由图可知，Indian Pines、Pavia University和KSC三个数据集合适的输入图像块半径分别为14.13和13。

表 14.11　三维卷积神经网络卷积流参数表

数　据　集	No.	Convolution	ReLU	Pooling	Dropout
Indian Pines	1	4 * 4 * 32 * 128	Yes	2 * 2	No
	2	5 * 5 * 32 * 192	Yes	2 * 2	50%
	3	4 * 4 * 32 * 256	Yes	No	50%
Pavia University	1	4 * 4 * 32 * 32	Yes	2 * 2	No
	2	5 * 5 * 32 * 64	Yes	2 * 2	50%
	3	4 * 4 * 32 * 128	Yes	No	50%
KSC	1	4 * 4 * 32 * 32	Yes	2 * 2	No
	2	5 * 5 * 32 * 64	Yes	2 * 2	50%
	3	4 * 4 * 32 * 128	Yes	No	50%

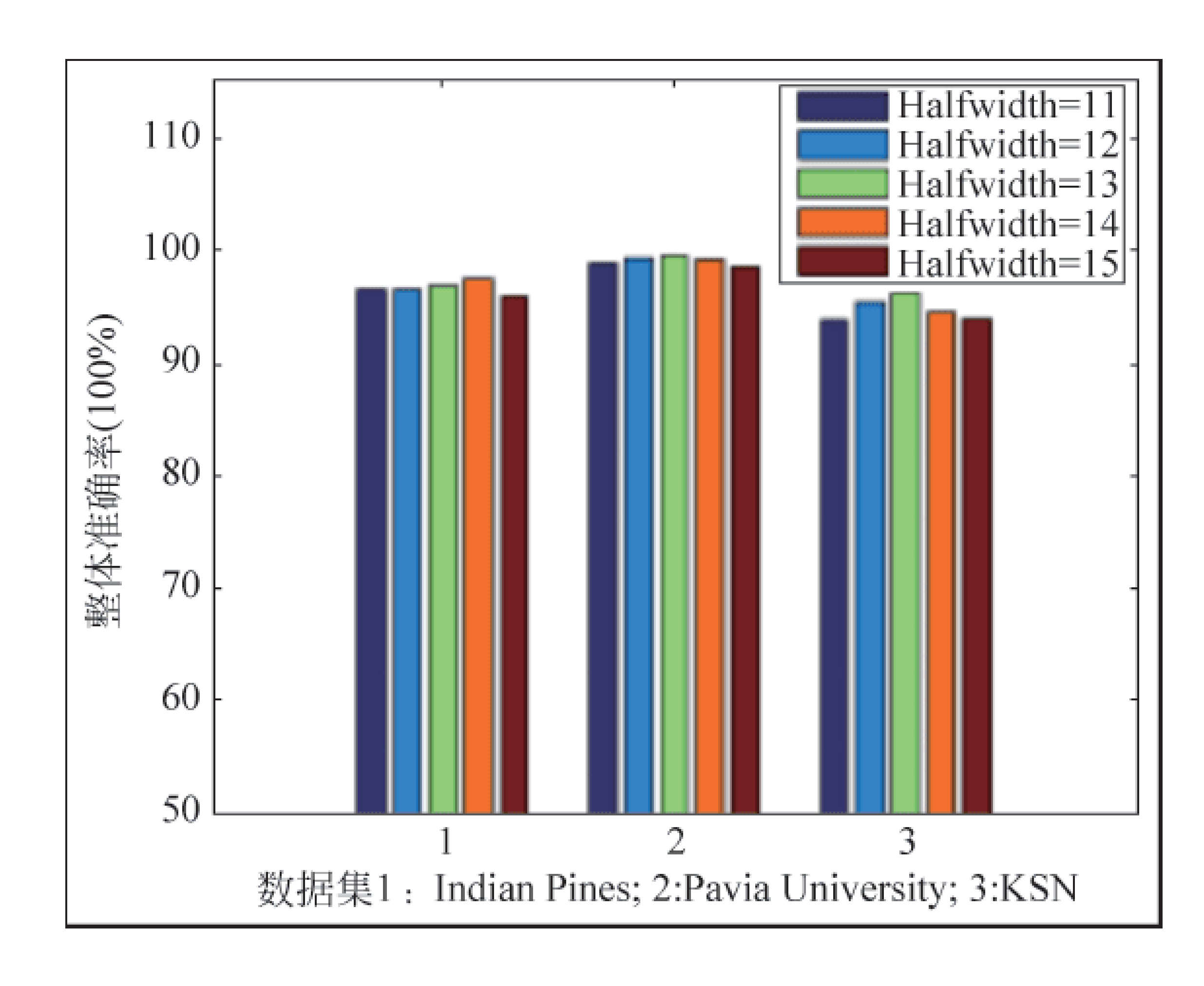


图 14.18　图像块半径对分类的影响

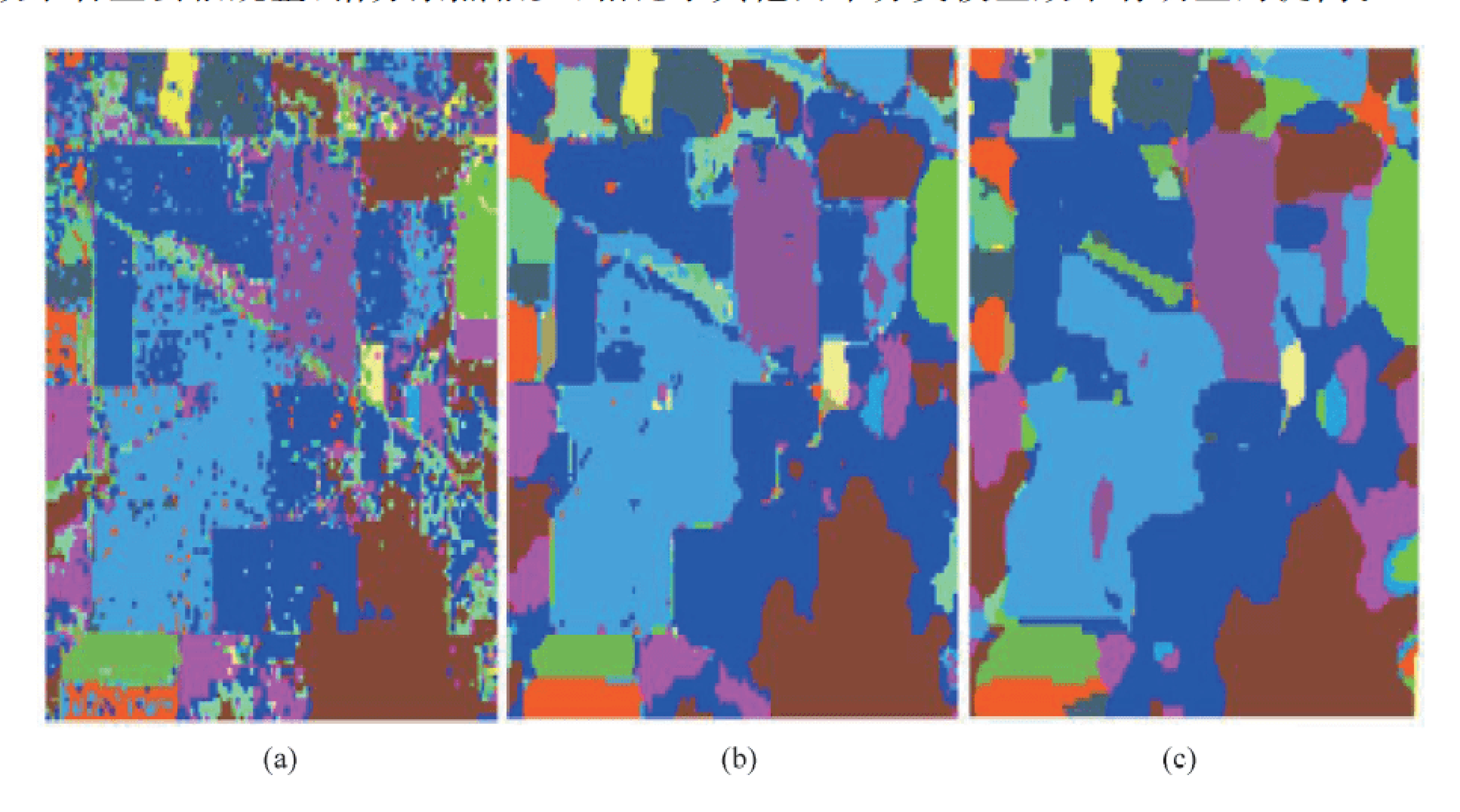
下一步我们将对比 3D-RBF-SVM、3D-EMP-RBF-SVM、3D-CNN-LR 三个模型在 Indian Pines、Pavia University、KSC 三个实验数据上的实验效果。本实验随机选取 20 组训练数据和测试数据，并在三个模型上进行测试，实验结果以“平均数±方差”的形式给出。OA、AA 和 Kappa 结果如表 14.12 所示。

由表 14.12 实验结果所示，相比于传统的支撑矢量机，基于空谱联合的三维卷积神经网络模型能够更好地提取空谱联合特征进行高光谱图像分类，在 OA、AA 和 Kappa 上显示出更好的结果。

表 14.12　三维卷积神经网络在三个数据集上的分类结果

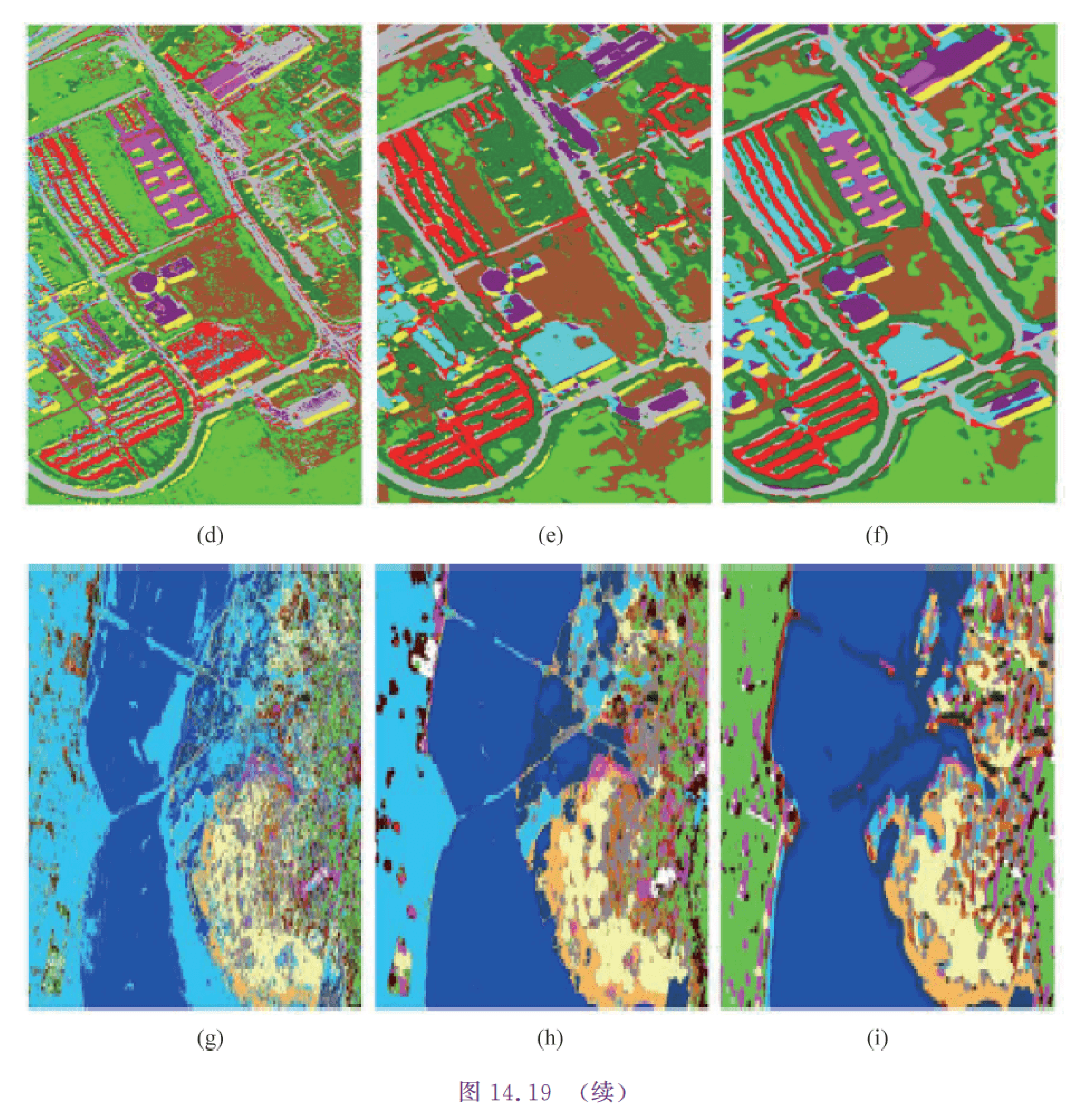
数　据　集	模　　型	3D-RBF-SVM	3D-EMP-RBF-SVM	3D-CNN-LR
Indian Pines	OA/%	92.42±1.10	96.92±0.81	97.56±0.43
	AA/%	92.14±1.44	95.07±0.88	99.23±0.19
	K×100	94.83±1.35	96.21±0.91	97.02±0.52
	Runtime/min	5.45	11.12	27.92
Pavia University	OA/%	96.05±0.91	97.72±0.61	99.54±0.11
	AA/%	95.73±0.61	97.17±0.47	99.66±0.11
	K×100	95.33±1.68	95.21±0.23	99.41±0.15
	Runtime/min	14.10	17.23	46.15
KSC	OA/%	93.73±0.67	95.66±0.61	96.31±1.25
	AA/%	90.55±1.52	93.82±0.93	94.68±1.97
	K×100	93.03±0.75	95.17±0.01	95.90±1.39
	Runtime/min	2.56	3.78	7.93

图 14.19 展示了三种模型在三个通用数据集上的分类效果图，由左到右分别为 1D-SVM、3D-EMP-RBF-SVM、3D-CNN-LR 对整张高光谱图像进行分类的结果。由结果可知，1D-SVM 分类结果存在很多嘈杂的乱点，错分现象很明显；3D-EMP-RBF-SVM 分类效果有所改善，错分杂点明显减少，但局部看上去还是有明显的错分现象；3D-CNN-LR 分类效果看上去很规整，错分杂点很少，相比于其他两个分类模型效果有明显的提高。



(a)　(b)　(c)

图 14.19　基于 1D-SVM、3D-EMP-RBF-SVM、3D-CNN-LR 模型的分类结果图



(d) (e) (f)

(g) (h) (i)

图 14.19 (续)

14.3 基于深度神经网络的高光谱影像的压缩

图像压缩技术是在保证图像重建质量前提下,用尽可能少的比特数表示图像数据中包含的信息。香农(Shannon)提出把数据看作信息和冗余的组合,数据冗余量与其可压缩程度成正比。若能大幅度去除冗余,则可实现有效压缩,降低描述图像所需的数据量。

传统的遥感图像压缩一般属于数据级压缩,只是为了减少存储空间与传输带宽,压缩数据不包含图像的特征。特征级压缩较之数据级压缩,不仅可以减少存储空间与传输带宽,而且可以有效减少图像分类、目标检测和识别等后续步骤的处理时间。目前,遥感图像的特征级压缩方法还很少,主要是基于字典学习的线性稀疏编码方法,线性稀疏编码是一种“浅层

架构”下的数据表征模型，只能学习到低级的诸如边缘方向的低阶特征，尽管可以通过字典学习获得稀疏描述空间，但通常不能挖掘出复杂非结构化场景的隐含解释性因素。

高光谱图像是一个三维的图像数据体，除了空间维图像外，还包括光谱维。由于光谱维的存在，高光谱图像的数据量远远超过普通的遥感图像，这就造成高光谱图像的数据获取难、传输难及存储难。故为了实现对其的应用，必须有效压缩高光谱图像的数据量，减轻数据的存储与传输负担。

14.3.1　基于深度自编码网络的高光谱图像压缩方法

我们提出了一种基于深度自编码网络的高光谱图像压缩方法。深度自编码网络是一种经典的深度学习模型，可自动、多层次提取高光谱图像的抽象高阶稀疏特征，属于特征级压缩方法。由于只需对压缩图像做简单的反量化和编码操作，减少了后期处理的时间，可用于高光谱图像的在轨实时大倍率压缩、存储与传输。

为了克服现有算法压缩前需要先进行去光谱间相关性的不足，深度自编码网络将高光谱图像所有波段的三维数据作为网络的输入，网络自动提取空间以及光谱维的特征，能够去除各波段间的光谱冗余以及波段内部的空间冗余。且图像压缩过程只需进行一次前向传递，其中仅涉及简单的矩阵乘法操作，因而实时性较好，实现简单，为在轨压缩提供了可能。

为挖掘复杂非结构化场景下的高光谱图像的高阶稀疏特征，利用深度学习思想构造深度自编码网络，深度自编码网络模型如图 14.20 所示。

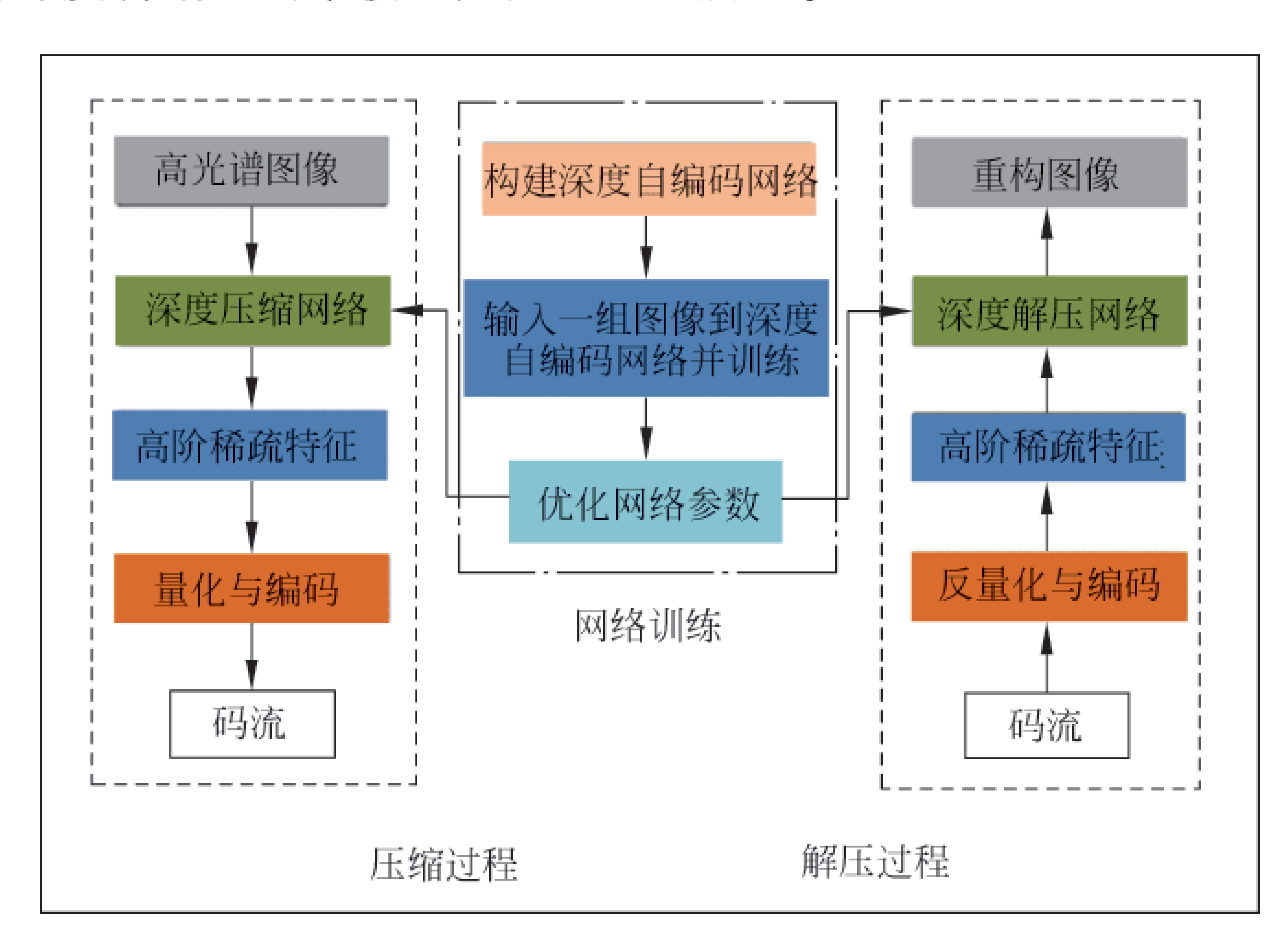


图 14.20　网络模型

深度自编码网络是一个由多个自编码器组成的神经网络，如图 14.21 所示，其前一层自编码器的输出作为后一层自编码器的输入，所以每个自编码器的输入层节点数目与隐藏层节点数目满足：后一个自编码器的输入层节点数等于前一个自编码器隐藏层的节点数，并

且第一个自编码器的输入层节点数等于原始输入的维度。

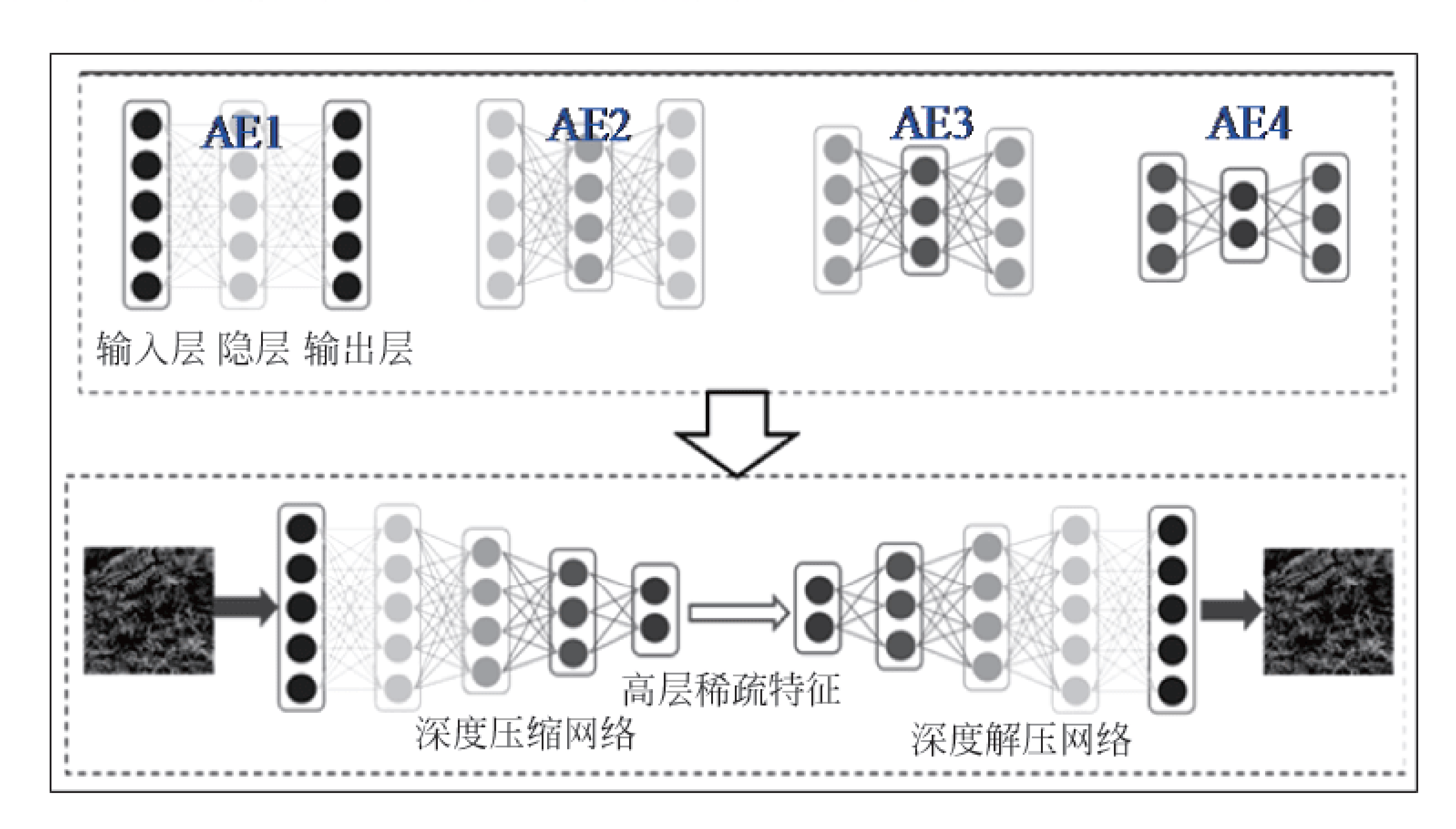


图 14.21 深度压缩网络与深度解压网络形成示意图

基于深度自编码网络的大压缩比高光谱图像压缩方法的具体步骤如下：

(1) 构建深度自编码网络，将多个自编码器级联堆叠构成深度自编码网络。

(2) 训练深度自编码网络，输入一组训练图像数据到深度自编码网络，训练该网络获得优化的网络参数，得到深度压缩网络和深度解压网络。训练好深度自编码网络，也就训练好了每个自编码器，然后按一定的规则堆叠这些自编码器构成深度压缩网络和深度解压网络。

(3) 将待压缩的高光谱图像送入深度压缩网络，计算网络各隐藏层的输出，得到层次化的逐步抽象的稀疏特征，对高阶稀疏特征进行量化和编码得到最终压缩码流，实现高光谱图像的大倍率压缩。

(4) 在深度解压网络中解压高光谱图像，对接收到的码流进行反量化和编码，得到高阶稀疏特征，将高阶稀疏特征送入深度解压网络，得到解压后的高光谱图像。解压后的高光谱图像通常也称为重构图像。

一般来说，峰值信噪比 PSNR 是传统的图像压缩算法性能应用最广泛的客观评价指标，通过计算误差的范数度量来描述重构图像与原始图像的失真。其计算公式如下：

$$\mathrm{PSNR}=10\cdot\log_{10}(V_{\mathrm{peak}}^2/\mathrm{MSE}) \tag{14.6}$$

$$\mathrm{MSE}=\frac{1}{M\times N}\sum_{i=1}^{M}\sum_{j=1}^{N}(f(i,j)-\hat{f}(i,j))^2 \tag{14.7}$$

其中 V_{peak} 为采用某种表示法所对应的峰值，如采用 8bit 表示法，对应峰值 $V_{\mathrm{peak}}=255$，$f(i,j)$ 为原始图像在 (i,j) 处的像素值，$\hat{f}(i,j)$ 为重构图像在 (i,j) 处的像素值，M、N 分别为图像的行数和列数。

14.3.2 实验设计及分类结果

深度自编码网络的训练数据库为 STL-10，此数据库包含 10 类物体，分别为 airplane、bird、car、cat、deer、dog、horse、monkey、ship、truck，每幅图像的大小为 96×96×3。可从如下网址下载：http://cs.stanford.edu/～acoates/stl10/。由于数据库较大（约2.65GB），故仅随机选取 1500 幅图像作为训练样本。

仿真实验的待压缩图像为一组干涉型高光谱图像，图像大小为 486×509×95，由于图像较大，实验时对图像进行了 8×8 的分块。采用的自编码器为稀疏自编码器，稀疏项惩罚因子 $\beta=0.005$，稀疏性参数 $\rho=0.05$；采用梯度下降法调整各自编码器网络参数，迭代次数 200 次。应用深度自编码网络对高光谱图像进行不同压缩倍率的压缩，不考虑量化与编码的压缩比分别为 8、16、32。本例中分别采用了 1 个、2 个、4 个自编码器，构成深度自编码网络并训练该网络，得到深度压缩网络和深度解压网络。硬件测试平台是：处理器 i5-3210M 2.5GHz，内存 4GB，软件平台为：Windows 8 64 位操作系统和 Matlab R2014a 64 位。

为了比较该网络在不同网络结构下的压缩与重构性能，将本例所述的高光谱图像在不同深度压缩网络结构下分别进行 8、16、32 倍的大倍率压缩，结果汇总至表 14.13。如果未标注，则训练数据数目为 1500 幅图像，即 648 000 个块；耗时一栏是指压缩高光谱图像每波段的平均耗时。

表 14.13 不同深度压缩网络结构下高光谱图像压缩 PSNR 值

深度压缩网络结构	压缩比	PSNR（dB，遥感，波段 10）	耗时（ms，每波段）
1(64-8)	8	28.43	23.16
1(64-4)	16	27.87	22.11
1(64-2)	32	14.76	21.05
2(64-48-8)	8	30.48	45.26
2(64-48-4)	16	29.19	41.05
2(64-48-2)	32	27.68	39.79
4(64-96-64-32-8)	8	27.52	93.28
4(64-96-64-32-4)	16	27.77	92.63
4(64-96-64-32-2)	32	26.74	91.58
4(64-96-64-32-4，2000 幅训练图像)	16	27.92	93.68

图 14.22(a)是一组待压缩的高光谱图像的第 10 波段的原始图像，图 14.22(b)是一组待压缩的高光谱图像的第 60 波段的原始图像；图 14.22(c)是采用 4 个自编码器构成的压缩比为 16 的深度压缩网络对图 14.22(a)所示图像压缩后重构结果图；图 14.22(d)是采用 4 个自编码器构成的压缩比为 16 的深度压缩网络对图 14.22(b)所示图像压缩后重构结果图。图 14.22 中(c)所示重构图像与图 14.22(a)所示原始图像间的 PSNR 为 27.77dB，图 14.22(d)所示重构图像与图 14.22(b)所示原始图像间的 PSNR 为 26.68dB．验证了深度自编码网络能以较高的质量压缩与重构卫星遥感影像。

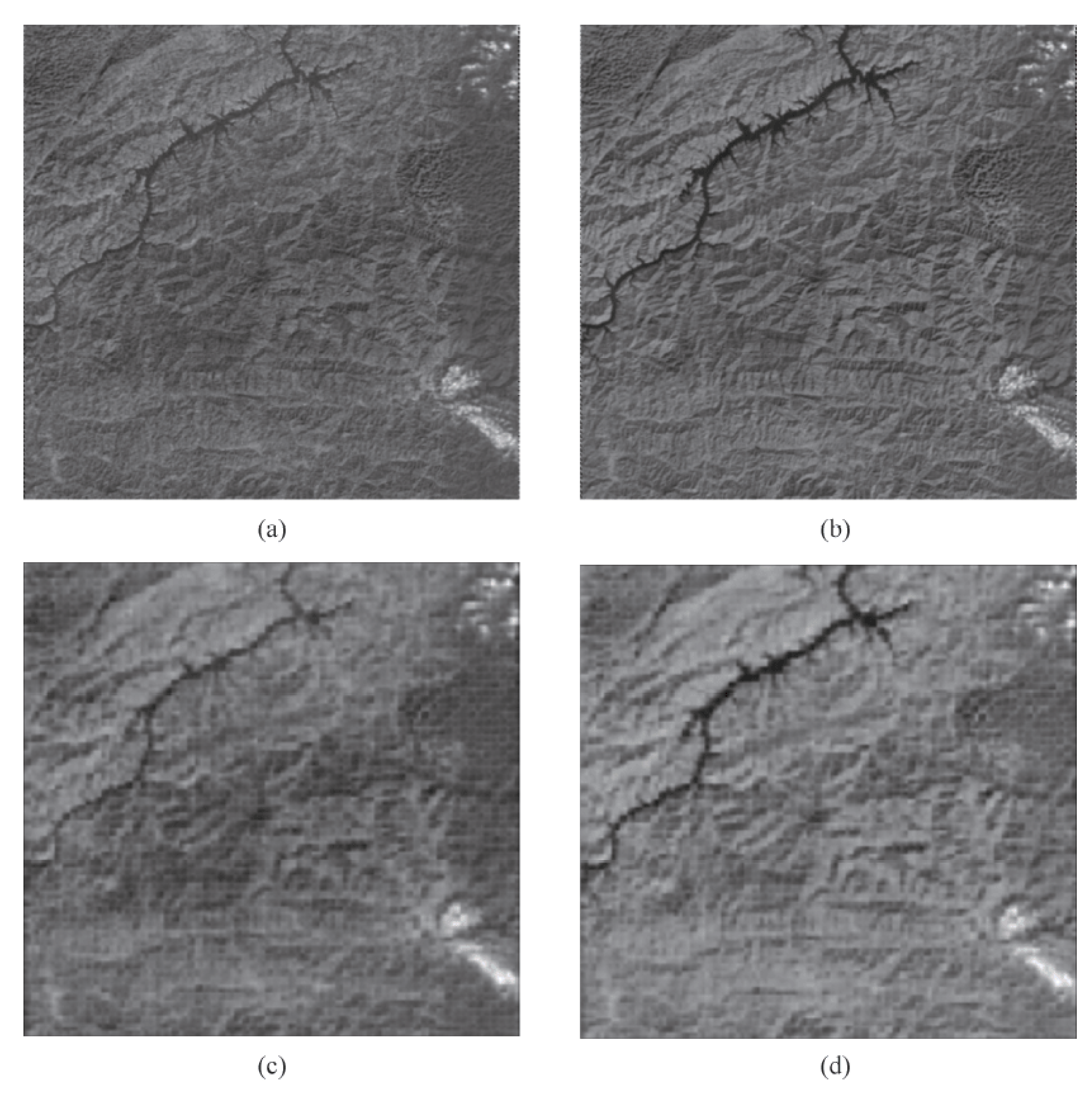


图 14.22　压缩重构图

从表 14.13 的 PSNR 客观评价结果可以看出，在相同压缩比的情况下：①深层的深度压缩网络能在保证图像重构质量的前提下实现更大的压缩比，即采用两个自编码器的深度压缩网络的压缩结果要优于采用一个自编码器的深度压缩网络的压缩结果，这是因为一个自编码器只能构成一个浅层的神经网络，学习到的只是数据的低级特征，不能得到高阶稀疏特征。②采用 4 个自编码器的深度压缩网络的压缩效果要次于采用两个自编码器的深度压缩网络的压缩效果，但从表 14.22 的最后一行可以看出，当我们增加训练 4 个自编码器构成的深度自编码网络的图像数据个数后，PSNR 会有所提高，这说明，更深层的网络需要更多的训练数据才能获得最优的表示。③本方法的实时性很好，为在轨压缩提供了可能，虽然网络结构越复杂，网络参数就越多，压缩高光谱图像所需的时间也越长，但基本上与网络结构复杂度呈线性关系，每个波段的图像压缩时间均不超过 0.1s，这是因为：采用本方法压缩高光谱图像，只需将图像输入深度压缩网络，进行一次简单的前向传递操作，涉及的运算多为矩阵乘法和加法。



参考文献

[14.1]　孙家柄. 遥感原理与应用[M]. 武汉：武汉大学出版社，2009.
[14.2]　理查兹. 遥感数字图像分析[M]. 北京：电子工业出版社，2009.
[14.3]　张良培. 高光谱遥感[M]. 北京：测绘出版社，2011.
[14.4]　万余庆. 高光谱遥感应用研究[M]. 北京：科学出版社，2006.
[14.5]　浦瑞良，宫鹏. 高光谱遥感及其应用[M]. 北京：高等教育出版社，2000.
[14.6]　童庆禧，张兵，郑芬兰. 高光谱遥感的多学科应用[M]. 北京：电子工业出版社，2006.
[14.7]　惠巍巍. 高光谱混合像元的分解及地物分类的研究[D]. 东北林业大学，2007.
[14.8]　关泽群. 遥感图像解译[M]. 武汉：武汉大学出版社，2007.
[14.9]　刘卓，易东云. 投影寻踪方法与高光谱遥感图像数据特征提取的研究[J]. 数学理论与应用，2003(1)：76-81.
[14.10]　Trizna D B，Bachmann C，Sletten M，et al. Projection pursuit classification methods applied to multiband polarimetric SAR imagery[J]. IEEE Transactions on Geoscience and Remote Sensing，2001,39(11)：2380-2386.
[14.11]　Zhang Y，Zhang J P，Jin M，et al. Adaptive subspace decomposition and classification for hyperspectral images[J]. Chinese Journal of Electronics，2000，9(1)：82-88.
[14.12]　Tropp J A，Gilbert A C，Strauss M J. Algorithms for simultaneous sparse approximation. Part I：Greedy pursuit[J]. Signal Processing，2006，86(3)：589-602.
[14.13]　Plaza A，Benediktsson J A，Boardman J W，et al. Recent advances in techniques for hyperspectral image processing[J]. Remote Sensing of Environment，2009，113(1)：S110-S122.
[14.14]　Licciardi G，Marpu P R，Chanussot J，et al. Linear Versus Nonlinear PCA for the Classification of Hyperspectral Data Based on the Extended Morphological Profiles[J]. Geoscience & Remote Sensing Letters IEEE，2012，9(3)：447-451.
[14.15]　Villa A，Benediktsson J A，Chanussot J，et al. Hyperspectral Image Classification With Independent Component Discriminant Analysis[J]. IEEE Transactions on Geoscience & Remote Sensing，2011，49(12)：4865-4876.
[14.16]　Waske B，Linden S V D，Benediktsson J A，et al. Sensitivity of support vector machines to random feature selection in classification of hyperspectral data[J]. IEEE Transactions on Geoscience & Remote Sensing，2010，48(7)：2880-2889.
[14.17]　Samat A，Du P，Liu S，et al. Ensemble Extreme Learning Machines for Hyperspectral Image Classification[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing，2014，7(4)：1060-1069.
[14.18]　Chen Y，Nasrabadi N M，Tran T D. Hyperspectral image classification via kernel sparse representation[C]. IEEE International Conference on Image Processing. IEEE，2011：1233-1236.
[14.19]　Zhou Y，Wei Y. Learning Hierarchical Spectral-Spatial Features for Hyperspectral Image Classification[J]. IEEE Transactions on Cybernetics，2015，46(7)：1667.
[14.20]　Li W，Prasad S，Fowler J E. Hyperspectral Image Classification Using Gaussian Mixture Models and Markov Random Fields[J]. Geoscience & Remote Sensing Letters IEEE，2014，11(1)：153-157.

[14.21] Chen Y, Lin Z, Zhao X, et al. Deep Learning-Based Classification of Hyperspectral Data[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing, 2014, 7(6): 2094-2107.

[14.22] Chen Y, Zhao X, Jia X. Spectral-Spatial Classification of Hyperspectral Data Based on Deep Belief Network[J]. IEEE Journal of Selected Topics in Applied Earth Observations & Remote Sensing, 2015, 8(6): 2381-2392.

[14.23] Yue J, Zhao W, Mao S, et al. Spectral-spatial classification of hyperspectral images using deep convolutional neural networks[J]. Remote Sensing Letters, 2015, 6(6): 468-477.

[14.24] Chen Y, Jiang H, Li C, et al. Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks[J]. IEEE Transactions on Geoscience & Remote Sensing, 2016, 54(10): 1-20.

[14.25] 赵学军. 高光谱图像压缩与融合技术[M]. 北京：北京邮电大学出版社，2015.

[14.26] 林洲汉. 基于自动编码机的高光谱图像特征提取及分类方法研究[D]. 哈尔滨工业大学，2014.

[14.27] Makantasis K, Karantzalos K, Doulamis A, et al. Deep supervised learning for hyperspectral data classification through convolutional neural networks[J]. 2015: 4959-4962.

[14.28] 戴晓爱，郭守恒，任清，等. 基于堆栈式稀疏自编码器的高光谱影像分类[J]. 电子科技大学学报，2016，45(3)：382-386.

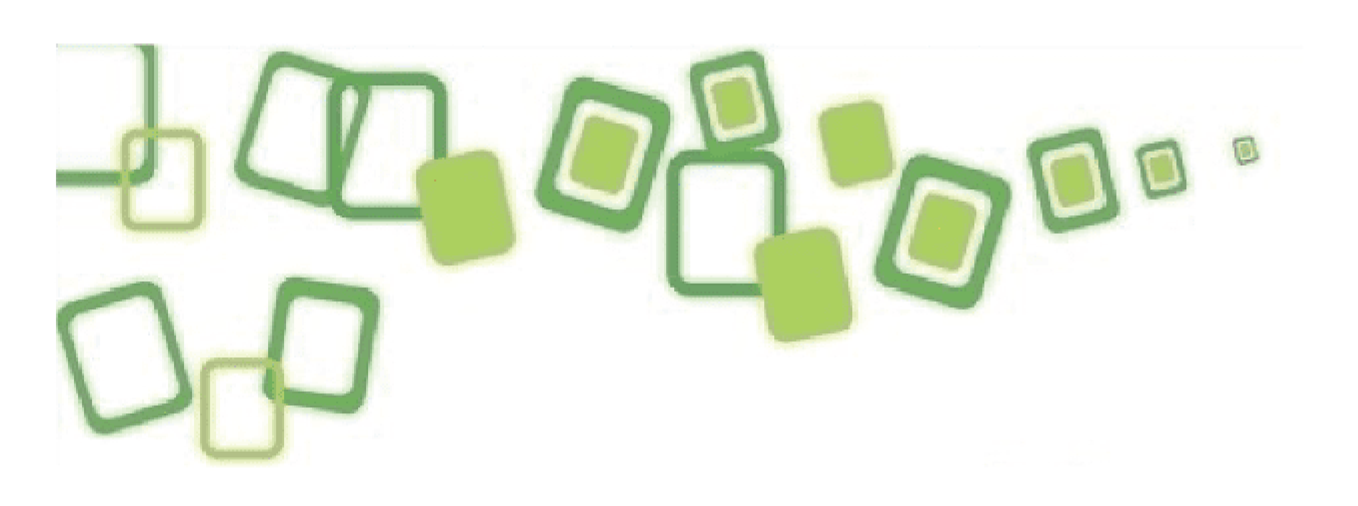
[14.29] 曹扬，赵慧洁，黄四牛，等. 基于高效置信传播的改进马尔可夫随机场高光谱数据分类算法[J]. 模式识别与人工智能，2014，27(3)：248-255.

[14.30] 张风. 基于子空间学习的高光谱影像地物分类[D]. 西安电子科技大学，2015.

[14.31] 李素婧. 面向大规模高光谱数据的半监督地物分类方法[D]. 西安电子科技大 学，2015.

[14.32] 周红静. 面向混合像元的高光谱遥感数据降维[D]. 西安电子科技大学，2015.

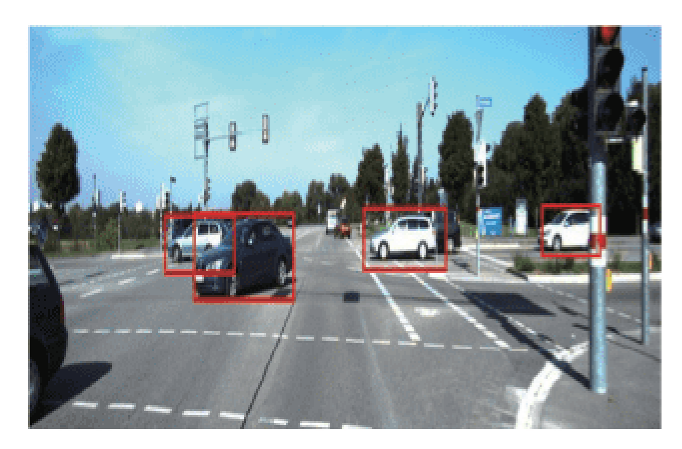
第15章

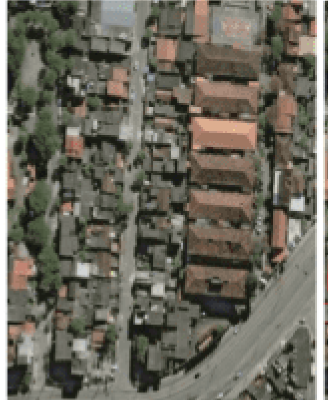


基于深度神经网络的目标检测与识别

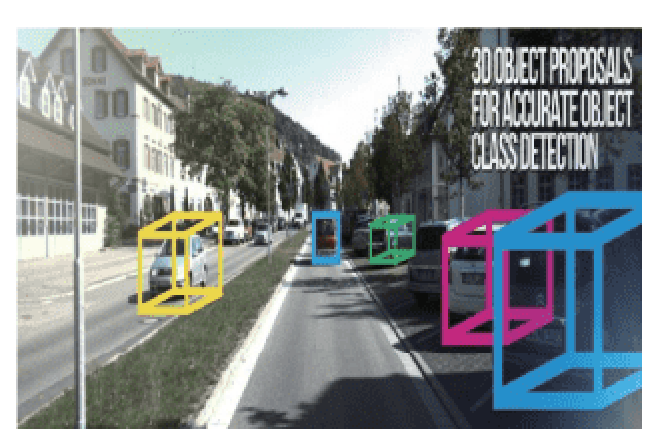
CHAPTER 15

目标检测与识别——深度神经网络

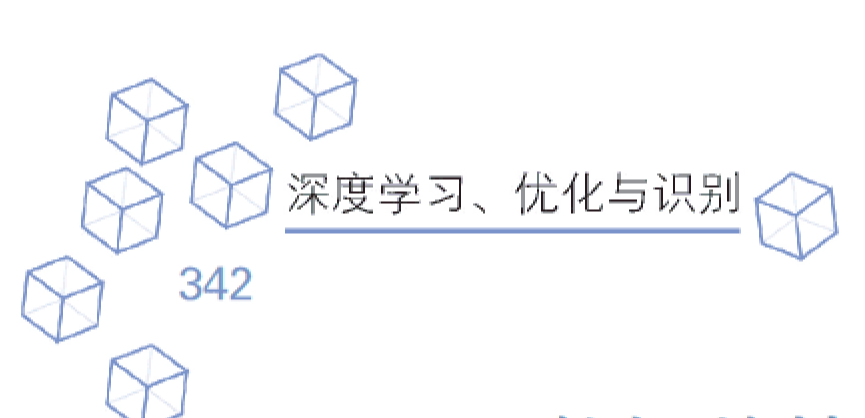








R-CNN SPP-Net Fast R-CNN Faster R-CNN R-FCN …



15.1 数据特性及研究目的

15.1.1 研究目的

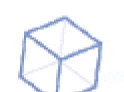
目标检测与识别在生活中的多个领域中有着广泛的应用，它是将图像或者视频中目标与其他不感兴趣的部分进行区分，判断是否存在目标，确定目标位置，识别目标种类的一种计算机视觉任务。目标检测与识别是计算机视觉领域中非常重要的一个研究方向。随着互联网、人工智能技术、智能硬件的迅猛发展，人类生活中存在着大量的图像和视频数据，这使得计算机视觉技术在人类生活中起到的作用越来越大，对计算机视觉的研究也越来越火热。目标检测与识别作为计算机视觉领域的基石，也越来越受到重视。在实际生活中应用也越来越广泛，例如：目标跟踪、视频监控、信息安全、自主驾驶、图像检索、医学图像分析、网络数据挖掘、无人机导航、遥感图像分析、国防系统等。

由于近年来目标检测与识别技术的火热发展，越来越多的相关成果发表在各种顶级期刊或者会议上，例如 *IEEE Transactions on Image Processing*（TIP），*Computer Vision and Image Understanding*（CVIU），*IEEE Transactions on Pattern Recognition and Machine Intelligence*（TPAMI），*Pattern Recognition*，*IEEE Transactions on Multimedia*，*International Journal of Computer Vision*（IJCV），*IEEE Conference on Computer Vision and Pattern Recognition*（CVPR），*International Conference on Computer Vision*（ICCV），*European Conference on Computer Vision*（ECCV），*ACM International Conference on Multimedia*（ACM MM）。在各国学者的共同努力下，目标检测与识别技术飞速发展，并使得最好的目标检测与识别算法在公开数据集上有着跨越式的进步，算法性能在不断地接近人类的能力。

就目标检测与识别技术的研究现状来说，可以将其分为两大类，分别为基于传统图像处理和机器学习算法的目标检测与识别方法和基于深度学习的目标检测与识别方法。

传统的目标检测与识别方法主要可以表示为：目标特征提取、目标识别、目标定位。这里所用的特征都是人为设计的特征，例如 SIFT、HOG 特征等。通过这些特征对目标进行识别，然后再结合相应策略对目标进行定位。这其中最著名的工作之一是 Felzenszwalb 团队在 2010 年提出的 DPM 模型。但这些传统的目标检测与识别方法在一些公开的数据集（PASCAL VOC）上并没有达到令人满意的结果。

现如今，基于深度学习的目标检测与识别方法已经成为主流，主要可以表示为：图像的深度特征提取和基于深度神经网络的目标识别和定位。其中用到的主要的深度学习模型是卷积神经网络（CNN）。2012 年 Hinton 教授的团队利用卷积神经网络设计了 AlexNet，使之在 ImageNet 问题上打败了所有传统方法的团队，使得 CNN 成为计算机视觉领域中最为重要的工具之一。这促使机器视觉研究进入了一个新的阶段。随后基于 CNN 的目标检测与识别方法也逐渐取代了传统方法。目前，可以将现有的基于深度学习的目标检测与识别



算法分为大致三类：基于区域建议的目标检测与识别算法，具有代表性的是 R-CNN、Fast R-CNN、Faster R-CNN；基于回归的目标检测与识别算法，具有代表性的是 YOLO、SSD；基于搜索的目标检测与识别算法，例如，基于视觉注意的 AttentionNet，基于强化学习的算法。

基于区域建议的目标检测与识别算法，这类算法的主要步骤是：首先通过，例如 Selective Search(SS)、Bing、EdgeBoxes 这些目标候选区域生成算法，生成一系列候选目标区域，然后通过深度神经网络提取目标候选区域的特征，并用这些特征进行分类，以及目标真实边界的回归。这种方式的目标检测识别算法占有非常大的比率。其中比较知名的就是 Ross Girshick 的 R-CNN 和 Fast R-CNN，以及 Shaoqing Ren 的 Faster R-CNN 这三种方法。R-CNN 可以说是开创性地将目标候选区域和深度学习相结合用来做目标检测，在当时是最好的目标检测算法。但由于 R-CNN 中单张图片所生成的候选区域过多，而且每判断一次候选区域，就要将图片进行区域提取然后再送入深度神经网络，使得这个算法效率不高。随后，Ross Girshick 在 R-CNN 的基础上提出了 Fast R-CNN 算法，这个算法不需要在输入图片上提取候选区域的图块，而是在图像对应的深度特征图上提取候选区域的特征图，并用这个特征图来做后续的识别和目标边界框回归。这两种方法的目标候选区域都是通过普通图像处理算法在原始图像上生成的，这使得算法在时间上的消耗比较大，为了改善这种缺陷，Shaoqing Ren 等人提出了 Faster R-CNN 算法，该算法将目标候选区域通过 Region Proposal Network(RPN)来生成，这使得整个目标检测和识别过程全部包含在一个深度神经网络内部，整个模型是一个端到端的过程，这样大大提高了算法的速度。当然这三种方法只是简单地对目标候选区域、目标的深度特征进行了处理。这些年也有各种各样基于 R-CNN 框架提出的改进算法。例如，Yuting Zhang 团队提出的通过贝叶斯优化和结构化预测的目标检测模型，文章中指出，当时的深度学习模型更易于判断目标类别，但是目标的定位仍然是个比较大的问题，因此提出了基于贝叶斯优化的搜索算法和一个结构化的损失函数来改善目标定位结果。Spyros Gidaris 团队提出的 MR-CNN & S-CNN & Loc 模型，就是从目标候选区域和目标深层特征图两方面来提高检测精度。首先将目标候选区域分成多种不同区域，这样可以更好地表达目标以及目标所在的环境，然后用 MR-CNN 提取这些区域的特征来共同表示这个候选区域，这一部分是对候选区域的深度利用。作者又设计了语义分割的 CNN 模型 S-CNN 来提取目标特征图上的前景特征，并将这些特征与 MR-CNN 提取的特征结合去做目标识别，最后再使用 CNN 来做目标位置的回归。Yukun Zhu 等人提出的 segDeepM 模型也是一种通过分割并结合目标区域的环境信息来增强目标检测与识别结果的方法。当然也有学者从深度特征和深度神经网络结构方面来优化目标检测结果，例如，Tao Kong 等人提出的 HyperNet 模型，这个模型就是通过将深度学习模型中个别层的特征图进行融合生成 Hyper 特征，依靠这个特征生成目标候选区域，然后提取目标候选区域的 Hyper 特征进行目标检测和识别。其他基于区域建议的目标检测与识别方法也有很多，例如 Wanli Ouyang 等人的 DeepID-Net，Sean Bell 等人的 Inside-Outside Net，Ning Zhang 等人的 Part-based R-CNNs，Jifeng Dai 等人的 R-FCN 模型等。

由于基于区域建议的目标检测与识别方法包含了各种各样的候选区域生成部分，以及



不同的特征层处理过程,使得这个算法的实时性得不到保证。因此,就催生了不生成目标候选区域而直接基于回归的目标检测与识别算法,主要包括 Joseph Redmon 等人提出的 YOLO 算法以及 Wei Liu 等人提出的 SSD 算法。YOLO 算法简单地将图片等分为多个部分,然后通过深度神经网络直接判断每一个部分是否存在目标,并预测目标类别和目标的边界框。这种方法不需要产生目标建议区域,节省了图像处理时间,使得检测的实时性得到了可靠的保证,最快能够做到每秒处理 155 帧图像。当然 YOLO 这种直接通过一个图像块来回归目标,在检测和识别精度上无法和基于区域建议的方法相比。因此,Wei Liu 等人提出的 SSD 就结合了 YOLO 中的回归思想和 Faster R-CNN 中的 anchor 机制,并使用不同特征图上的多层特征去检测和识别图像中不同位置的目标。这样既保证了检测和识别有较快的速度,又保证了结果有和基于区域建议的目标检测与识别方法相近的精度。

基于搜索的目标检测与识别算法也是近两年逐渐出现的,它主要是用一种自顶向下的搜索策略在整个图像上搜索目标,然后识别搜索到的结果,可以分为基于强化学习的目标检测,例如 Juan C. Caicedo 等人的方法是设计了 9 种动作,通过深度神经网络和深度 Q 学习来预测每次要执行的动作,再根据动作来搜索目标。Míriam Bellver Bueno 等人同样提出用强化学习来搜索目标,与 Juan C. Caicedo 不同的是,Míriam Bellver Bueno 等人只用了 6 种动作来搜索目标,然后通过将图像分层表示来逐渐缩小目标搜索范围,这个搜索过程同样用到了深度学习和深度强化学习中的 Q 学习策略,最终达到检测和识别的目的。也可以利用视觉注意的机理来实现基于搜索的目标检测和识别方法,例如,Donggeun Yoo 等人提出的 AttentionNet 模型,这个方法与强化学习预测目标搜索动作比较类似,这里直接通过深度神经网络来预测视觉注意的趋势,根据预测出来的趋势不断地趋向真实目标位置,然后识别对应目标。

这些就是对现有主流的基于深度学习的目标检测与识别方法的简单概括,后续章节将会对上述 3 种方式中比较有代表性的模型做详细的介绍。

15.1.2 常用数据集

对于目标检测与识别任务,国际上有各种公开的数据集,其中最具代表性的是 PASCAL VOC 数据、COCO 数据、ImageNet 数据。这些数据都是彩色自然图像,基本上都是来源于网络。

1. ImageNet 数据集

ImageNet 是一个用于图像分类/定位/检测的常见数据集。包含 1400 多万的海量图像数据,有着 1000 个目标类别,如图 15.1 所示。其中超过百万的图片有着明确的类别和位置信息的标注,可以下载的数据包括原始图像、SIFT 特征、目标边框和目标属性等。ImageNet 数据是目前世界上图像识别最大的数据集,在深度学习和图像领域的发展中有着重要的意义,很多论文在研究中使用到了该数据集。基于 ImageNet 的竞赛 Large Scale Visual Recognition Challenge(ILSVRC)推动了计算机视觉识别挑战的持续发展。随着深度学习的发展,近些年来,中国团队也在该竞赛中屡屡获得奖项。

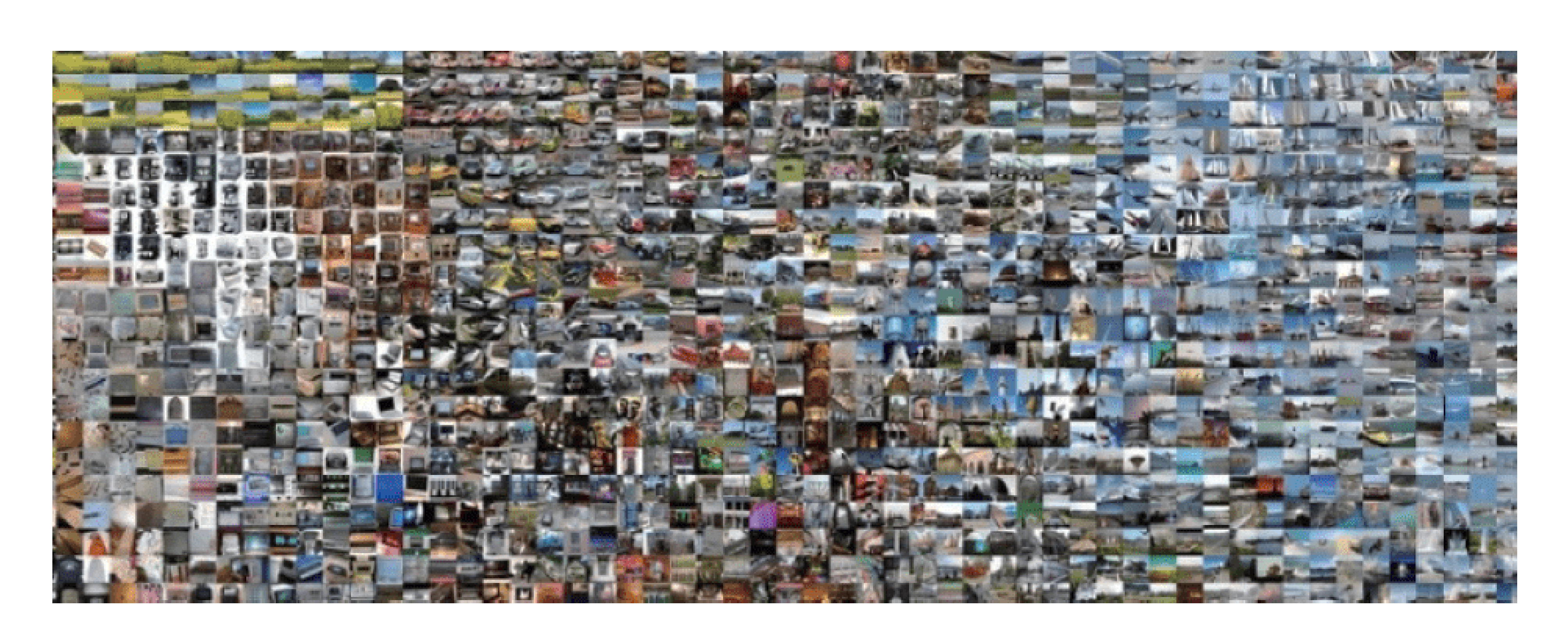


图 15.1　ImageNet 数据集

ImageNet 的具体信息如下：

- 图像总类别：21841。
- 图像总数：14 197 122。
- 标注目标的图像数量：1 034 908。
- 根据 SIFT 划分的图像类别：1000。
- 有 SIFT 特征的图像数量：1.2million。
- 数据集地址：http://www.image-net.org/index。

2. COCO 数据集

COCO(Common Objects in COntext)是由微软公司赞助的一个新的用于图像识别/图像分割/语义标注的数据集，图像的标注信息包括类别/位置和语义文本描述。COCO 数据集收集的是自然环境下的包含常见对象的日常场景，有超过 30 万幅图片，250 万目标标记，如图 15.2 所示。COCO 数据集中包含 91 个常见的对象类别，其中有超过 5000 个标记的类别有 82 个，每张图片中有着不止一个目标。COCO 数据集分为三个部分：训练集 165 482 例，验证集 81 208 例，测试集 81 434 例。同样，基于 COCO 数据集的图像识别挑战赛 MS COCO(Microsoft Common Objects in COntext challenges)也吸引了各界的关注和参与。

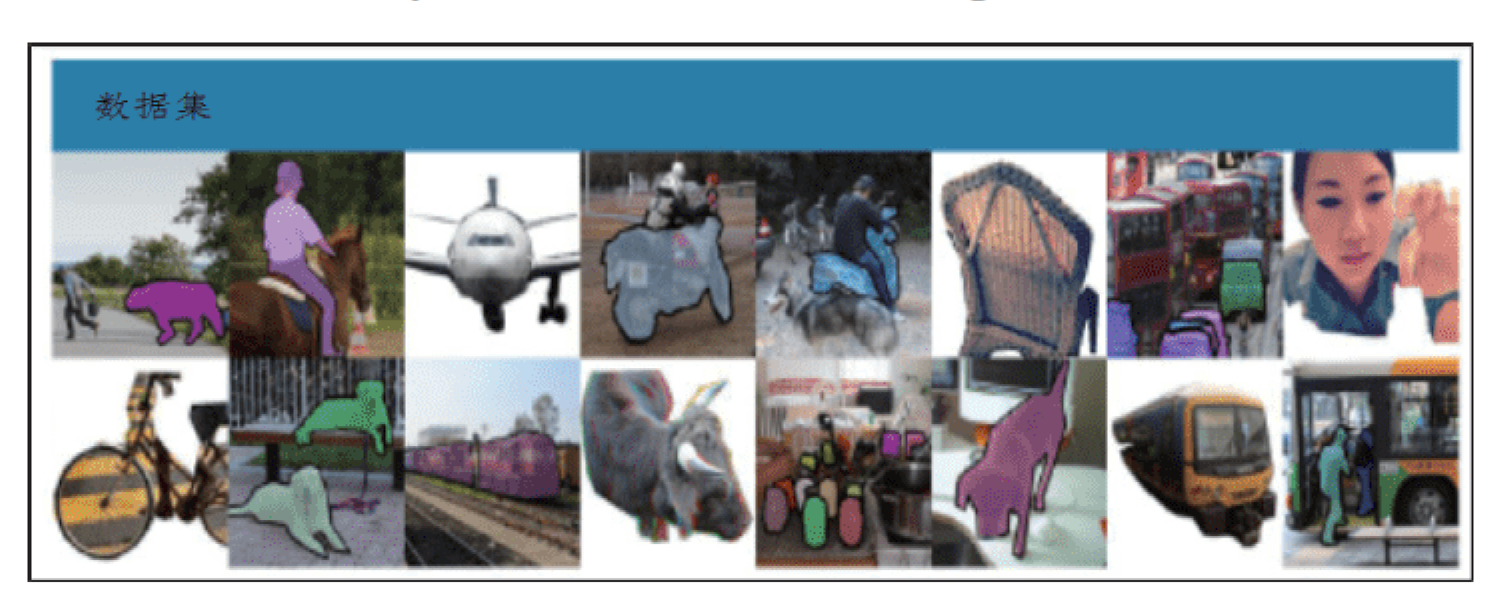


图 15.2　COCO 数据集

备注：COCO 数据集地址链接为 http://mscoco.org/.

3. VOC 数据集(图 15.3)

PASCAL VOC(Pattern Analysis Statistical modelling, Computational and Learning Visual Object Classes) 数据集是一个用于图像分类/识别/分割的数据集。PASCAL VOC 中有一万多幅图片，有 20 个目标类别，如图 15.3 所示。分别是人类，动物(鸟、猫、牛、狗、马、羊)，交通工具(飞机、自行车、船、公共汽车、小轿车、摩托车、火车)，室内(瓶子、椅子、餐桌、盆栽植物、沙发、电视)。采用这些在生活中常见的目标类别，可以更好地体现算法在实际应用中的实用性。VOC 数据集包括训练集/验证集/测试集三个部分，标注信息以 XML 形式保存。PASCAL VOC 挑战赛从 2005 年开始，到 2012 年结束。虽然比赛不再进行，但是 VOC 数据集图像质量高，标注完善，仍被很多人研究采用，是计算机视觉领域的重要数据集之一。

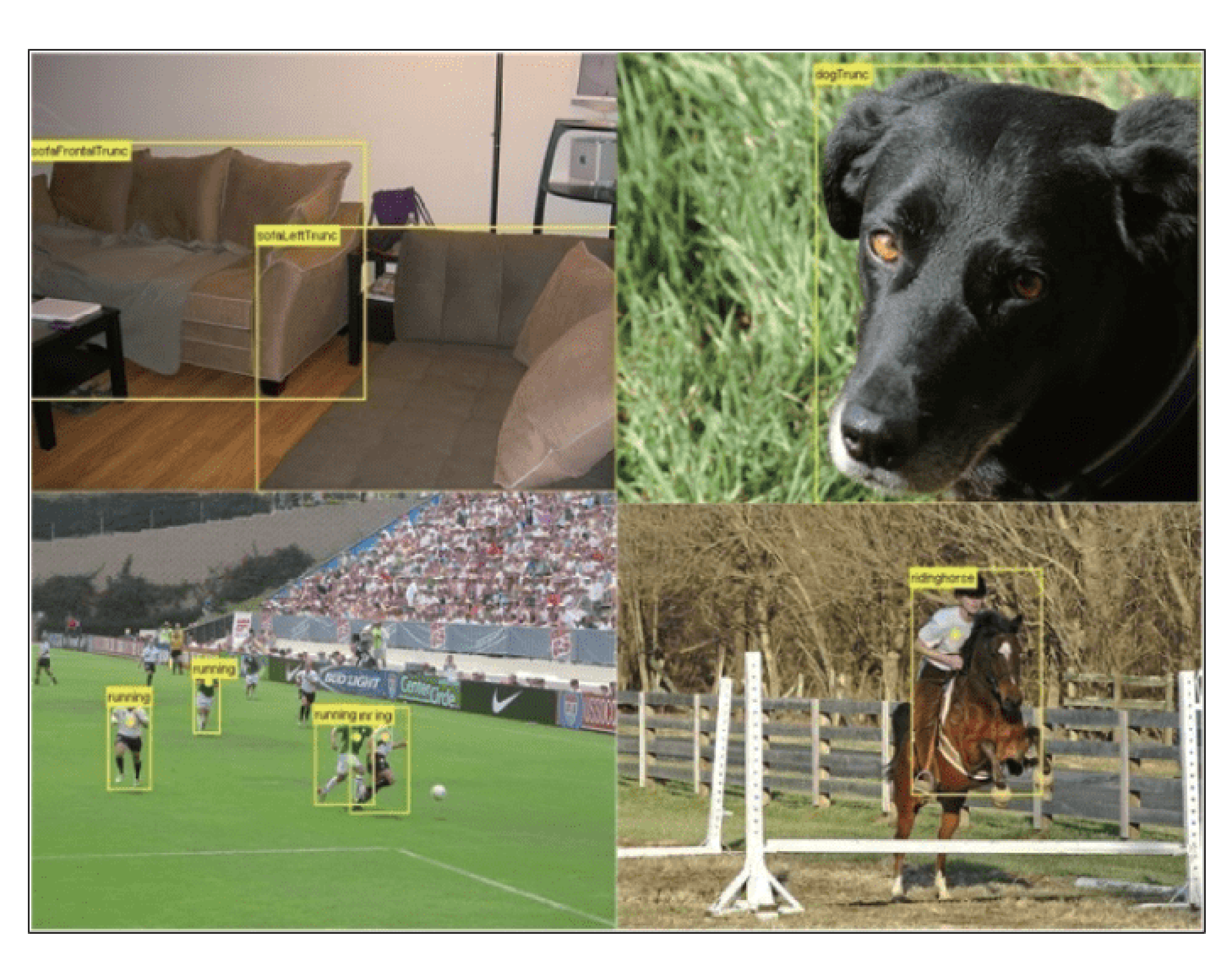


图 15.3　VOC 数据集

备注：VOC 数据集地址链接为 http://host.robots.ox.ac.uk/pascal/VOC/index.html.

15.2　基于快速 CNN 的目标检测与识别

基于区域选择的目标检测与识别算法是一种现阶段最成熟、应用最为广泛的目标检测与识别框架。它将整个检测识别过程简化成一个分类任务，并利用深度学习方法在大规模复杂数据分类上的优越性能来提升检测精度，这种框架一经出现就打败了同时段出现的其他所有目标检测方法。经过各国学者的不断研究，极大地提升了这种框架下的目标检测识



别精度以及算法的实时性。本节将讲述几种典型的结合深度学习的基于区域选择的目标检测与识别方法。

15.2.1 R-CNN

在 Ross Girshick 等人提出 DPM 方法之后，目标检测进入了一个瓶颈期，复杂的特征提取和集成学习等方式也只能得到极其有限的提升，人工特征在目标检测上起到的作用越来越有限。而深度学习的出现带来了新的曙光，Ross Girshick 等人在深度学习的浪潮下提出了一种基于深度学习的目标检测与识别方法——R-CNN. Ross Girshick 将目标检测与识别任务划分为基于候选区域提取的分类任务，就这样非常有效地利用了深度学习在分类任务上的强大性能，开创了利用深度学习进行目标检测的时代。

相较于传统方法，R-CNN 主要的优势有：①利用的不再是人为设计的特征，而是通过深度学习方法得到的更具表达能力的深度特征，提高了整个任务的识别精度；②采用区域建议的方式提取可能目标，而不是用滑窗的方式去检测目标，这样能够减少很多不必要的识别过程；③加入了边界框回归的策略来进一步提高检测精度。

当然仍然有不足的地方，例如，用了区域建议的方法，对每个建议区域都要重新计算整个网络，使得运算效率不高，也没有将区域建议过程融合在整个深度学习模型中，无法做到端到端的处理任务。

整个 R-CNN 模型可以用一句话来概括，首先用选择性搜索算法(Selective Search，SS)提取目标候选区域，通过深度 CNN 网络提取每一个候选区域的深度特征，训练 SVM 分类器来对这些特征进行分类，最后通过边界框回归算法重新定位目标边界框。整个算法的框架可以用文中示意图来表示，如图 15.4 所示。

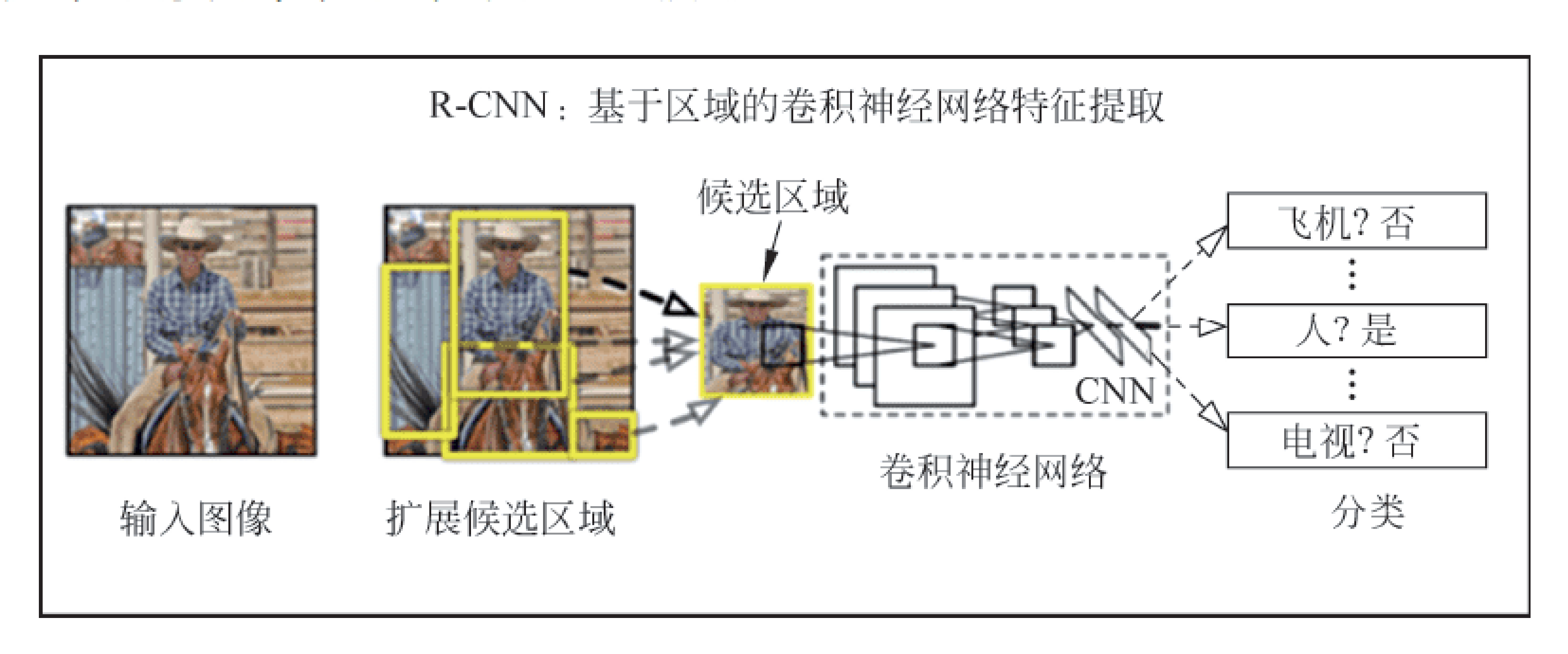


图 15.4 R-CNN 模型整体框架

R-CNN 模型中首先要做的就是候选区提取，在经典的目标检测算法中，使用的是滑窗法来进行候选区域的提取。在每张图片中，需要生成大量的候选区，有的甚至多达上百万。而在 R-CNN 中，使用了选择性搜索(Selective Search)，预先提取一些(约为 1～2k)可能包含物体的候选区域，随后对这些区域进行进一步的特征提取，判断是否为目标，这样大大提



升了算法的效率。

随后 R-CNN 对每一个候选区域做特征提取，该过程使用了深度网络，这里网络的输入就是经过大小调整的候选区域图像。网络设计上参考了 2012 年 Hinton 在 Image Net 上的分类网络（A Krizhevsky，I Sutskever，G Hinton. ImageNet classification with deep convolutional neural networks），当然现在也可以用最后提出的一些其他的基础深度学习模型，例如 VGG。由于 CNN 需要的是固定大小的图片作为输入（Alex-net 的输入大小为 227×227），而目标候选区的大小并不同。因此在输入提取特征的 CNN 网络之前，需要将候选区裁剪为 227×227 大小。在论文中，作者也讨论了几种不同的裁剪方式。通过特征提取的 CNN 网络，每一个特征候选区得到一个 4096 维的特征向量。

得到深度特征后就要依据这些特征来识别每个候选区域。这里使用的分类器为 SVM，输入为 CNN 网络提取的每个候选区的特征，输出为每个候选区的类别。在 PASCAL VOC 2007 中，类别数为 20 类＋背景类。在 ILSVRC 2013 中，类别数为 200 类＋背景类。

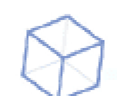
当然目标候选区域并不一定完全准确，因此这里还需要做边框回归（Bounding-Box Regression），这样能使目标的定位更为准确。目标检测的性能衡量标准之一是预测边框与目标边框之间的 IOU 指数(两边框的交集比并集)。如果一个预测边框与目标边框的 IOU 太小，那么这个目标依然相当于没有检测到。因此，在 R-CNN 中引入了边框回归的策略。这个边框回归在本质上和 DPM 中的边框回归是一样的。最后就是通过非最大抑制策略来删减不必要的边框。

目标检测的标注数据通常是比较少的。但是图片分类却有着大量的有标记数据。实验表明，将一个任务训练好的参数作为另一个网络的初始化参数，相比于随机初始化的参数，精度可以有很大的提高。因此 Ross Girshick 使用了预训练加精调的方法。ILSVRC 2012 数据库是一个包含大量有标记图片的数据库。R-CNN 首先使用了 ILSVRC2012 的全部数据对网络进行预训练（也可以直接使用 AlexNet 的网络和训练参数），最后得到的是一个 1000 维的类别输出。

由于 PASCAL VOC 2007 中的数据为 20 个类别，因此将预训练后的网络最后一层由 1000 替换为 21（包括 20 个类别和 1 个背景类）的全连接层，然后使用 PASCAL VOC 2007 中的全部数据对网络进行微调。目标和背景通过候选框和标定框之间的 IOU 来选取。重叠比例大于 0.5 时，标定为目标；小于 0.5 时，标定为背景。在每一次迭代训练中都使用 32 个正样本（包括所有类别）和 96 个背景样本组成的 128 张图片的 batch 进行训练。精调使用的算法为随机梯度下降（SGD）。

在测试阶段，首先通过 SS 得到 1～2k 个目标候选区。将每个候选区缩放到固定大小 227×227 后送入 CNN 网络进行特征提取，得到每个目标候选区的特征表示。该特征为一个 4096 维的向量，将每个特征向量输入到 SVM 分类器中进行分类，判断该目标候选区为背景或者目标以及相应的目标类别。对得到的目标候选区进行非极大值抑制（NMS）以及边框回归，得到更加准确的目标定位。

R-CNN 使得目标检测研究得到了突破性的进展，在 PASCAL VOC 将准确率从



35.1%提升到53.7%。但是R-CNN也有着一定的局限性。目标候选区的重叠使得CNN特征提取的计算中有着很大的冗余,这在很大程度上限制了检测速度。针对这一缺点,提出了R-CNN的升级版本Fast R-CNN。

15.2.2 Fast R-CNN

虽然R-CNN在检测识别任务上突破了传统方法的限制,但是它仍然存在各种各样的问题,因此Ross Girshick再次提出了Fast R-CNN模型,主要用于解决R-CNN的三个问题:①整个模型分为多个步骤,包括Selective Search提取的目标候选区,训练CNN特征提取模型,训练SVM分类器,训练边框回归器;②测试时间长,由于每张图片要处理大量目标候选框;③训练时所需空间大,花费时间多,R-CNN在训练时每个候选区域都要调整成相同大小的图像,并输入到网络中,这使得处理一张图片所需的空间大,整个模型需要训练CNN模型、SVM分类器以及目标边框回归器,所以训练时间花费很多。

对比于R-CNN,Fast R-CNN在目标候选区域生成方面没有改变,同样用到了SS策略。不同的是Fast R-CNN提出了Regions of Interest(RoIs)策略将候选区域映射到CNN模型的特征层上,直接在特征层上提取对应区域的深层特征,避免了不断输入不同区域图像的部分。然后将提取到的特征直接用Softmax预测区域类别,用网络来学习一个边界框回归器。将整个特征提取,分类和边界回归都整理成一个部分,提高了整个模型的效率。该模型可以用图15.5来表示。

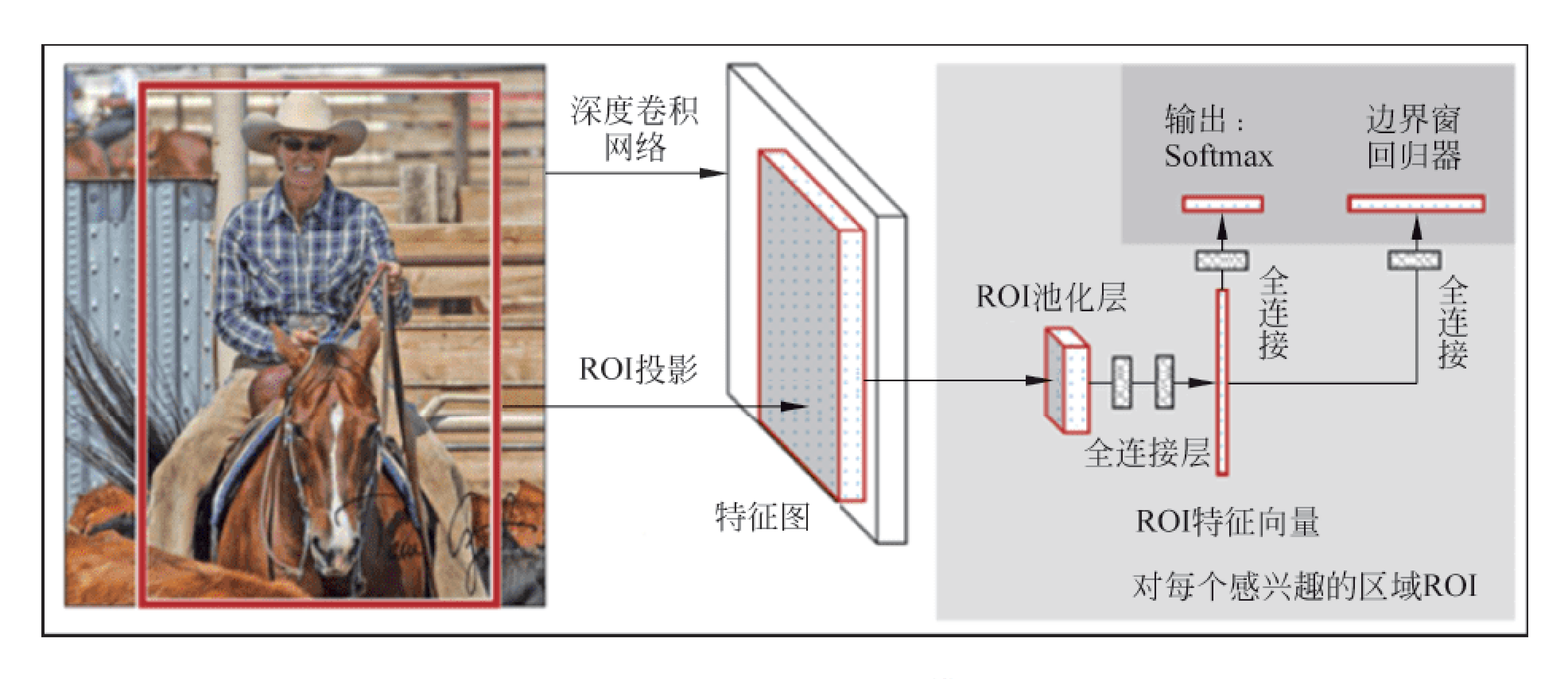
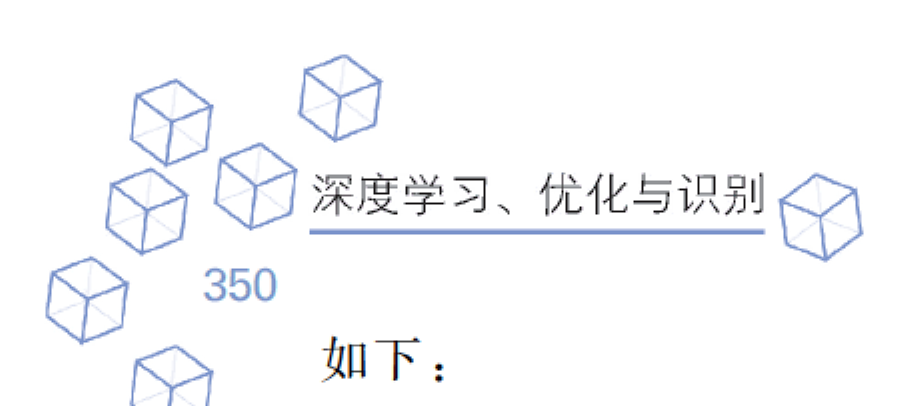


图15.5 Fast R-CNN模型

Fast R-CNN有两个平行的输出层。分类层输出的是每个RoI在$k+1$个类别上的概率分布$\boldsymbol{p}=(p_0,\cdots,p_k)$。边界回归层输出的是边框回归的参数,$\boldsymbol{t}^k=(t_x^k,t_y^k,t_w^k,t_h^k)$,其中$k$表示类别。这两个输出层是通过一个联合损失函数来训练的:

$$L(\boldsymbol{p},\boldsymbol{u},\boldsymbol{t}^u,\boldsymbol{v}) = L_{\text{cls}}(\boldsymbol{p},\boldsymbol{u}) + \lambda[\boldsymbol{u} \geqslant 1] \cdot L_{\text{loc}}(\boldsymbol{t}^u,\boldsymbol{v}) \tag{15.1}$$

这里,$L_{\text{cls}}(\boldsymbol{p},\boldsymbol{u})=-\log(\boldsymbol{p}_u)$是真实类别$\boldsymbol{u}$的对数损失。而回归损失$L_{\text{loc}}$的定义基于两组参数:类别$\boldsymbol{u}$的真实边框$\boldsymbol{v}=(v_x,v_y,v_w,v_h)$,类别$\boldsymbol{u}$的预测边框$\boldsymbol{t}^u=(t_x^u,t_y^u,t_w^u,t_h^u)$,详细表述



如下：

$$t_x = \frac{(G_x - P_x)}{P_w} \tag{15.2}$$

$$t_y = \frac{(G_y - P_y)}{P_h} \tag{15.3}$$

$$t_w = \log\left(\frac{G_w}{P_w}\right) \tag{15.4}$$

$$t_h = \log\left(\frac{G_h}{P_h}\right) \tag{15.5}$$

其中(G_x,G_y,G_w,G_h)表示真实目标的中心坐标和边框宽和高，(P_x,P_y,P_w,P_h)表示候选区域的中心坐标和区域的宽和高。对于边框回归层，定义的损失为：

$$L_{\text{loc}}(\boldsymbol{t}^u,\boldsymbol{v}) = \sum_{i\in\{x,y,w,h\}} \text{Smooth}_{L_i}(t_i^u - v_i) \tag{15.6}$$

其中：

$$\text{Smooth}_{L_i}(x) = \begin{cases} 0.5x^2 & |x| < 1 \\ |x| - 0.5 & \text{其他} \end{cases} \tag{15.7}$$

与 R-CNN 中的 L_2 范数相比，Fast R-CNN 中使用的 L_1 范数鲁棒性更强。

Fast R-CNN 在训练时首先用 ImageNet 数据集预训练整个 VGG 网络。然后用 VOC 目标检测数据集调整整个网络，并训练网络最后部分的分类器和边界框回归器。在 Fast R-CNN 中，使用 SGD 来进行训练，每次选择 2 幅图片，128 个候选区，并将每张图片定为固定尺寸。也就是说，每次训练只输入两幅整幅图片，但是可以进行 128 个 RoI 的训练，每 64 个 RoI 之间都可以共享特征计算。这种训练方式大大提高了训练效率。而且在训练过程中只对数据做了镜像，而没有采用其他的方式来做数据增强。

所以，从主要框架中可以看出，Fast R-CNN 是将整幅图像输入到网络，在网络正向传播训练过程中提取候选区域。而且并不需要将候选区的特征向量保存起来，将类别判断和位置精调学习统一起来，并不是像 R-CNN 是多个阶段训练。所以不仅节省时间，而且也不需要过大的磁盘存储。但是 Fast R-CNN 仍然用到了候选区域生成算法，需要识别的区域仍然很多，而且这部分算法暂时没有办法融入 GPU，所以在一定程度上影响了整个算法的效率。

15.2.3 Faster R-CNN

Fast R-CNN 克服了 R-CNN 提取卷积特征时冗余操作的缺点，将目标检测的特征提取，分类和边框回归统一到了一个框架中。然而，目标候选区域提取步骤仍然独立于整个深度神经网络单独存在，Fast R-CNN 和 R-CNN 中使用的目标候选区提取方法 SS 耗时相对较长，且难以融入 GPU 运算，于是目标候选区提取成为限制检测速度的一个新的瓶颈。

为了解决了这一问题，Shaoqing Ren 等人提出了 Faster R-CNN 算法，该算法中引入了一个新的概念——区域生成网络(Region Proposal Networks，RPN)来进行目标候选区的提取。从某种意义上讲，Faster R-CNN 可以看作是由生成目标候选区的 RPN 和利用这些候选区的 Fast R-CNN 检测器组成的，这样整个目标候选区域提取、深度特征提取、目标识别和检测过程都融入在一个深度神经网络模型中。这样所有过程都可以在 GPU 中运行，从而大大提高了整个检测速度却不降低检测精度。

与 Fast R-CNN 和 R-CNN 不同的是，Faster R-CNN 模型只需要输入一张图片，以及图片中目标的类别和对应的边界框类别。随后通过基础的 CNN 模型对图像做特征提取，这里用的是 VGG 网络，并且只用到了前 13 层卷积网络。然后将输出的特征用 RPN 做候选区域的预测，以及用预测到的候选区域边框对特征图做如同 Fast R-CNN 中 RoI 的操作，并达到目标的识别和边界框回归。整个过程可以用图 15.6 表示。其中 RPN 是一个全卷积神经网络(Fully Convolutional Network，FCN)，其输入前一层为任意大小的特征图，输出为一系列的矩形目标候选区。为了生成候选区域，一个小型网络在共享卷积网络的最后一层卷积层的输出特征图上进行了滑窗选择。该网络的输入为特征图的一个 $n \times n$ 的窗口。对于每个窗口，同时预测 k 个目标候选区，这 k 个候选区都与这个窗口存在关联，称为 anchors。每个 anchor 都有着对应的尺度和比例。卷积特征图中的每一个点都是一个 anchor 中心，有着 k 个相对应的 anchors。对于一张 $w \times h$ 大小的卷积特征图，存在 $w \times h$ 个 anchors。每个窗口被映射为一个低维的向量(在 VGG-16 网络中为 512 维)。该特征向量随即被传送到两个子网络中：边框回归网络和边框分类网络。边框分类网络输出的是每个 anchor 属于目标或者背景的概率，对于每个窗口，有 $2k$ 个输出，即将 256 维向量映射为 $2k$ 维向量；而边框回归网络输出的是每个 anchor 的平移缩放的值，对每个窗口，有 $4k$ 个输出。整个 RPN 如图 15.7 所示。

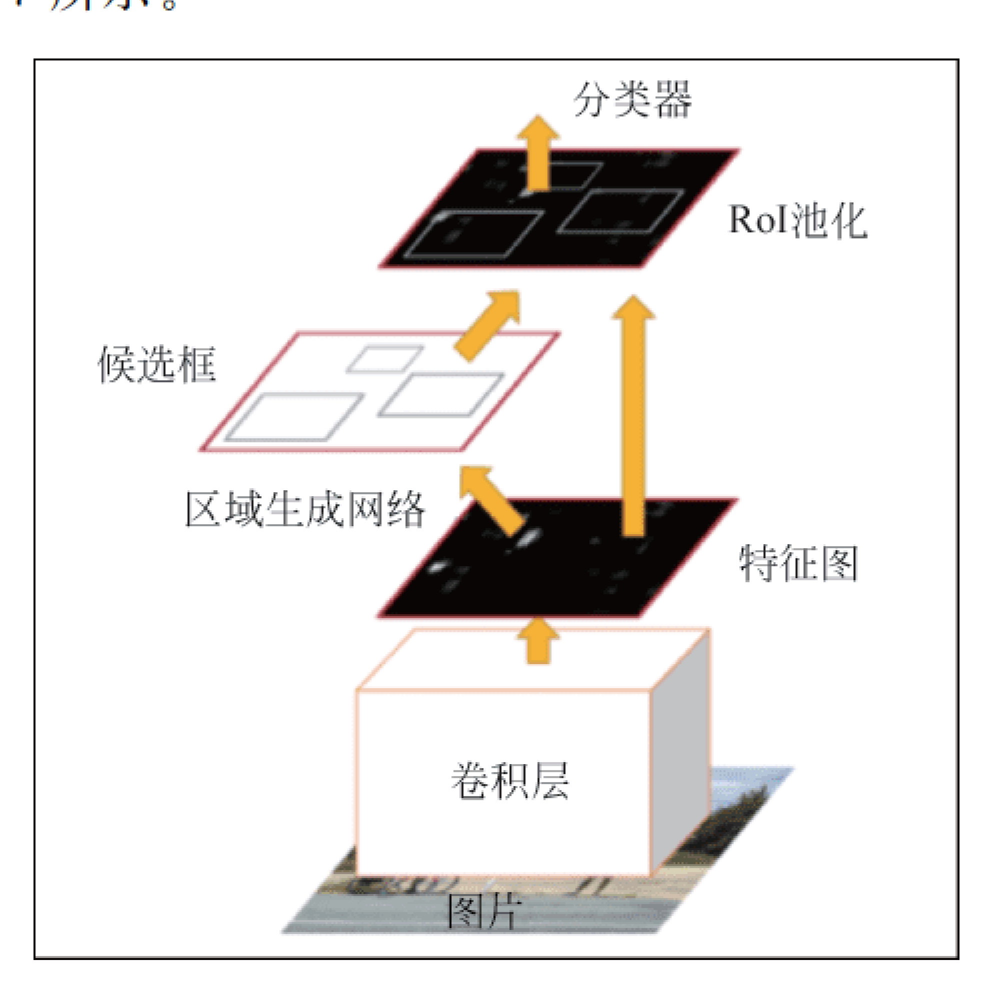


图 15.6　Faster R-CNN 模型框架

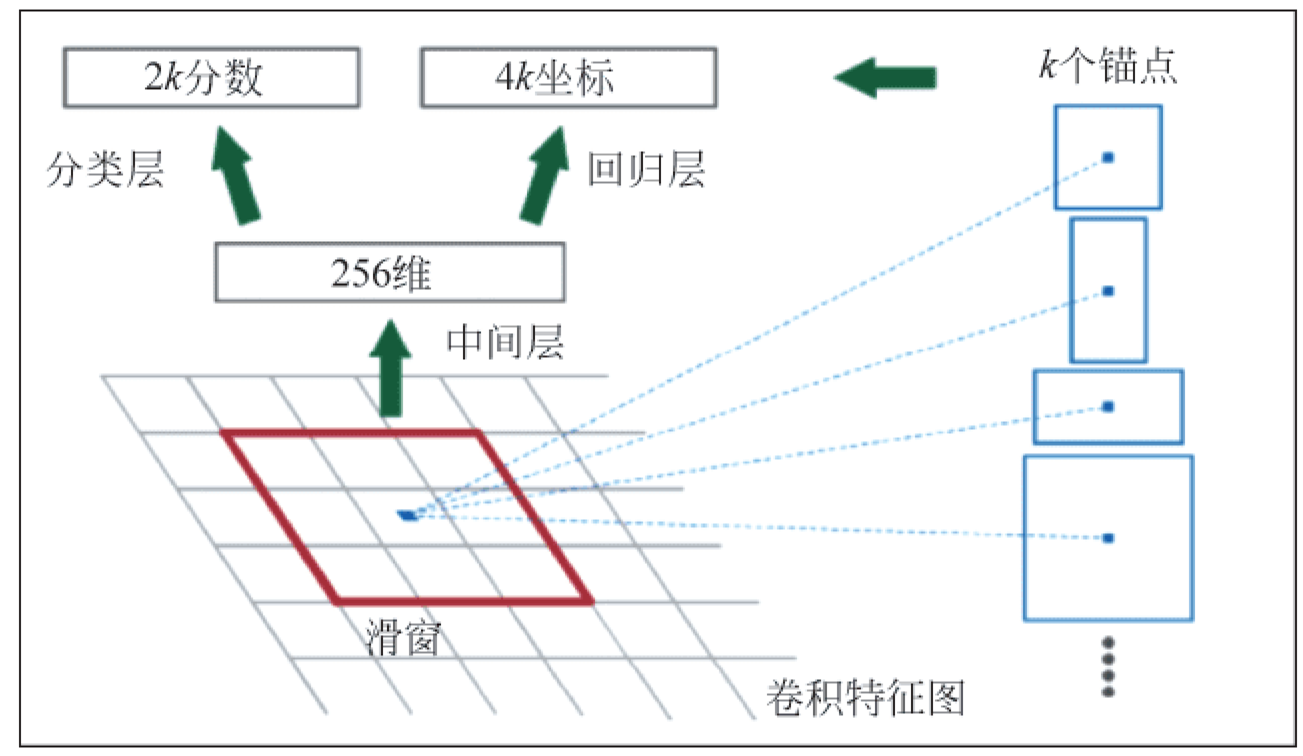


图 15.7　Region Proposal Network(RPN)网络模型

RPN 的训练过程是端到端(end-to-end)的。使用的优化方法是反向传播(back-propagation)和随机梯度下降(SGD),损失函数是分类误差和回归误差的联合损失:

$$L(\{p_i\}+\{t_i\})=\frac{1}{N_{\mathrm{cls}}}\cdot\sum_i L_{\mathrm{cls}}(p_i,p_i^*)+\frac{\lambda}{N_{\mathrm{reg}}}\cdot\sum_i p_i^*\cdot L_{\mathrm{reg}}(t_i,t_i^*)\qquad(15.8)$$

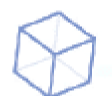
其中 i 表示第 i 个 anchor 点,$p_i^*=1$ 表示第 i 个 anchor 点为正样本,t_i^* 表示候选区域边框和真实目标边框之间的偏差。在训练时,正样本为与任意的真实边框(ground truth)的 IoU 大于 0.7 的候选区;负样本为与所有的 ground truth 的 IoU 小于 0.3 的候选区。而不属于以上两种情况的目标候选区不在训练中使用。

通过 RPN 得到候选区域后,将得到的区域作用于 VGG 的特征图,进行与 Fast R-CNN 相似的 RoI Pooling 操作,提取对应区域的特征,并用这些特征去做目标类别和边界框的预测。这部分的损失函数与 Fast R-CNN 的基本一样。若整个模型去掉 RPN 部分,剩下在网络中的部分与 Fast R-CNN 一样。

RPN 和 Faster R-CNN 共享了一个提取特征的卷积神经网络。为了精调这个网络,Faster R-CNN 提出了三种方法。

- 交替训练:先训练 RPN,然后用 RPN 上提取的候选区域训练 Fast R-CNN,接着在 Fast R-CNN 的参数基础上训练 RPN,周而复始,迭代训练。
- 近似联合训练:在每次迭代中,前向传播时将 ROI 视为固定的,在反向传播时将 RPN 的损失和 Fast R-CNN 的损失合并。
- 非近似联合训练:在训练中,考虑 ROI 区域变化的影响。作者采用了交替训练的方法来训练 Faster R-CNN 模型。

整个模型提出了 RPN 来生成候选区域,将整个检测识别过程完全封装到一个深度学习模型中,整个过程可以完全在 GPU 中运行,在测试时提高了检测速度,使得它在效率上完全打败了 R-CNN 和 Fast R-CNN。



15.2.4　对比实验结果与分析

这里将展示本节介绍的基于区域建议和深度学习的目标检测与识别方法的实验结果。并与一些传统的不使用深度学习模型的目标检测与识别方法做对比。这里采用了VOC2007测试数据集来对比各种方法的结果，如表15.1所示。

表15.1　VOC2007测试数据检测结果(%)

Detectors	DPM	R-CNN	Fast R-CNN	Faster R-CNN
mAP	33.7	66.0	70.0	73.2
aero	33.2	73.4	77.0	76.5
bike	60.3	77.0	78.1	79.0
bird	10.2	63.4	69.3	70.9
boat	16.1	45.4	59.4	65.5
bottle	27.3	44.6	38.3	52.1
bus	54.3	75.1	81.6	83.1
car	58.2	78.1	78.6	84.7
cat	23.0	79.8	86.7	86.4
chair	20.0	40.5	42.8	52.0
cow	24.1	73.7	78.8	81.9
table	26.7	62.2	68.9	65.7
dog	12.7	79.4	84.7	84.8
horse	58.1	78.1	82.0	84.6
mbike	48.2	73.1	76.6	77.5
person	43.2	64.2	69.9	76.7
plant	12.0	35.6	31.8	38.8
sheep	21.1	66.8	70.1	73.6
sofa	36.1	67.2	74.8	73.9
train	46.0	70.4	80.4	83.0
tv	43.5	71.1	70.4	72.6

由表15.1可以看出，传统的目标检测与识别方法在mAP指数上无法和基于深度学习的方法相比较。R-CNN在PASCAL VOC2007测试数据上将准确率从33.7%提高到了66.0%，提高了接近1倍，实现了巨大的飞跃，这得益于CNN模型在分类任务上的巨大成功。而基于区域建议的检测方法通过引入目标候选区域，将目标检测与识别任务化简为一个分类任务，完美地利用了CNN的强大性能。从各类的mAP指数上可以发现DPM这种传统方法没有任何优势。基于区域建议的三种方法在各类目标的检测结果上都有很大的提高。从R-CNN、Fast R-CNN、Faster R-CNN这三种方法的结果可以看出，经过不断的改进，基于区域建议的方法精度越来越高。但是可以看出bottle、chair、plant这几种目标的检测结果仍然不太令人满意。这三种目标在数据中都是比较小的目标，而且类内的形态多种



多样。想要分好这些目标则需要更好的目标候选区域提取方法,更好的深度学习模型,更多的训练数据。

在时间消耗上,DPM 算法处理每张图片大概需要 2 秒,R-CNN 大概 47 秒处理一张图片,Fast R-CNN 大概需要 3 秒处理一张图片,Faster R-CNN 每秒可以处理大概 5 张图片。由于 R-CNN 用了 SS 算法,并且识别和边界回归是分开处理的,因此整个时间复杂度是最高的。Fast R-CNN 仍然由 SS 算法做预处理,但是通过 RoI 策略将区域特征提取放入网络内部,并且直接通过深度神经网络来做识别和边界回归,因此提高了算法的效率。Faster R-CNN 去掉了 SS 预处理过程,用 RPN 网络自己生成有用的目标候选区域,然后用 Fast R-CNN 的后续过程来检测识别目标,整个过程完全在深度神经网络内部进行,形成一种端到端的模型,效率是最高的。

15.3 基于回归学习的目标检测与识别

基于区域建议的目标检测与识别方法由于存在候选区域提取,使得算法会花费更多的时间,很难做到实时检测和识别,而且检测结果也受到候选区域提取算法的影响。而基于回归的目标检测与识别算法由于没有候选区域提取步骤,且所有识别和检测步骤都可以融合在一个深度神经网络中处理,因此很容易做到实时检测和识别。但是,一般基于相同基础深度学习模型,比如 VGG 网络时,基于区域建议的模型会有更令人满意的检测和识别结果,而基于回归的目标检测识别模型在时间效率上更好。后续内容将详细介绍 YOLO 和 SSD 算法。

15.3.1 YOLO

对于人类来说观察一张图片,检测并识别出图像中的目标是非常快速、准确的,而且不需要反复观察一张图片。因此 Joseph Redmon 等人提出了 YOLO(You Only Look Once)算法,这个算法对于基于区域建议算法来说,是一种新的目标检测与识别框架。他们将目标检测与识别当作一种回归问题,通过回归的方式来检测目标的位置以及识别目标的类别。而且以回归的方式只需要用单一的网络对整张图片做一次评估就可以得到目标边界框和类别。这个算法是一个端到端的模型,因此这个算法可以说只要看图片一次就能检测和识别目标。

YOLO 算法较传统方法有如下几个优点:①这个方法足够快,整个模型的框架非常简单,在 Titan X GPU 下能做到每秒 45 帧,快速版的能做到每秒 150 帧,可以做到视频的目标检测与识别。②YOLO 每次能够直接检测和识别到整张图的所有目标,每次能够处理整张图,这样在识别目标时相当于加入了目标所在的周围环境的信息。③YOLO 能够学习到目标的一种概括性的表示,当 YOLO 可以检测和识别自然图像中的人时,它同样可以用来检测艺术品中的人物。

YOLO 算法也有如下几个缺点：①定位精度差，该算法的定位精度没有基于区域建议的算法高，这是由于该算法只是简单地对图像做回归导致的；②对一些小目标或者目标之间位置很接近的情况下检测效果不好，这也是因为该算法只是对固定大小、固定位置的图像块做回归引起的。

整个 YOLO 算法可以分为以下三步：①将整个图片等分成 $S\times S$ 个格子；②将整张图片送入深度神经网络，预测每一个格子是否存在目标、目标的边界框、目标的类别；③将预测的边界框做非最大抑制(NMS)筛选出最好的边界框，从而得到最好的结果。这里用 YOLO 文献中的示意图来表示这个过程，如图 15.8 所示。

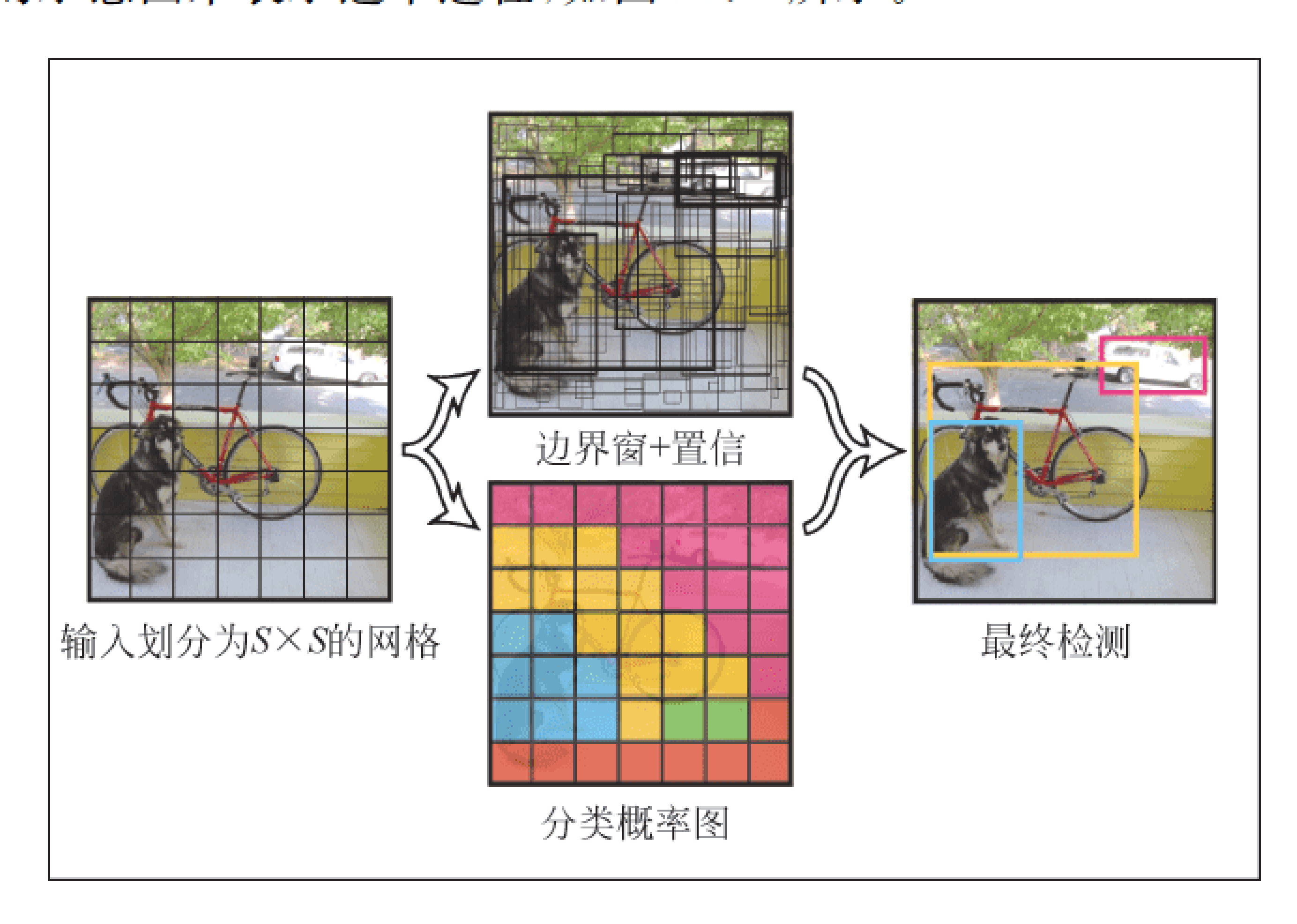


图 15.8 YOLO 整体框架

训练时每次运行整个模型需要用到的数据包括图片 Im 以及图片中目标的类别$\hat{c}$和目标边界框坐标$(\hat{x},\hat{y},\hat{w},\hat{h})$，其中$(\hat{x},\hat{y})$表示真实目标的中心位置坐标，$(\hat{w},\hat{h})$表示真实目标边界框的宽和高。整个系统在处理检测识别任务时，首先将整个图片划分成 $S\times S$ 个格子，如图 15.8 中所示。这样划分图像非常简单而且能够确保格子中可能存在目标。文中 $S=7$，这样整个过程只需要判断 49 个部分是否存在目标，并预测目标位置。这比基于区域建议的方法动辄判断几百上千的区域要简单快捷，直接提高了整个过程的效率。最后对整张图片目标位置类别的预测可以用一个 $S\times S\times(B\times 5+C)$的张量来表示。其中 B 表示图像中每个格子要预测的目标边界框的数量，文中 $B=2$，表示一次预测两个边界框。这里的 5 表示目标边界框(x,y,w,h)和这个格子对于目标的置信度分数 Conf，共 5 个参数。C 表示所用数据集中目标的类别数量，如果用的是 PASCAL VOC 数据，则 $C=20$。因此，最终预测一个 $7\times 7\times 30$ 的张量。文中设计的目标置信度分数 Conf 用来表示图像中格子预测的边界框是否包含目标，以及预测的边界框是否准确。定义如下：

$$\text{Conf}=P(\text{Class}_i \mid \text{Object})\cdot P(\text{Object})\cdot \text{Iou}_{\text{pred}}^{\text{truth}}=P(\text{Class}_i)\cdot \text{Iou}_{\text{pred}}^{\text{truth}} \tag{15.9}$$

其中 $P(\text{Class}_i|\text{Object})$表示有目标的情况下，目标属于第 i 类的概率，$P(\text{Object})$表示边界

框中是否有目标，$\text{Iou}_{\text{pred}}^{\text{truth}}$表示预测的目标边框和真实的目标边框之间的交集与并集的比值。这个参数将在测试阶段评价筛选出来的边框是否有效。

第二步就是设计深度神经网络来提取图像特征，并预测目标类别和边界框。文中设计的网络结构是基于 GoogLeNet 改进的，这是一种基于卷积神经网络(Convolutional Neural Network，CNN)搭建的深度神经网络。它的结构可以引用文中的图示来表示，如图 15.9 所示。

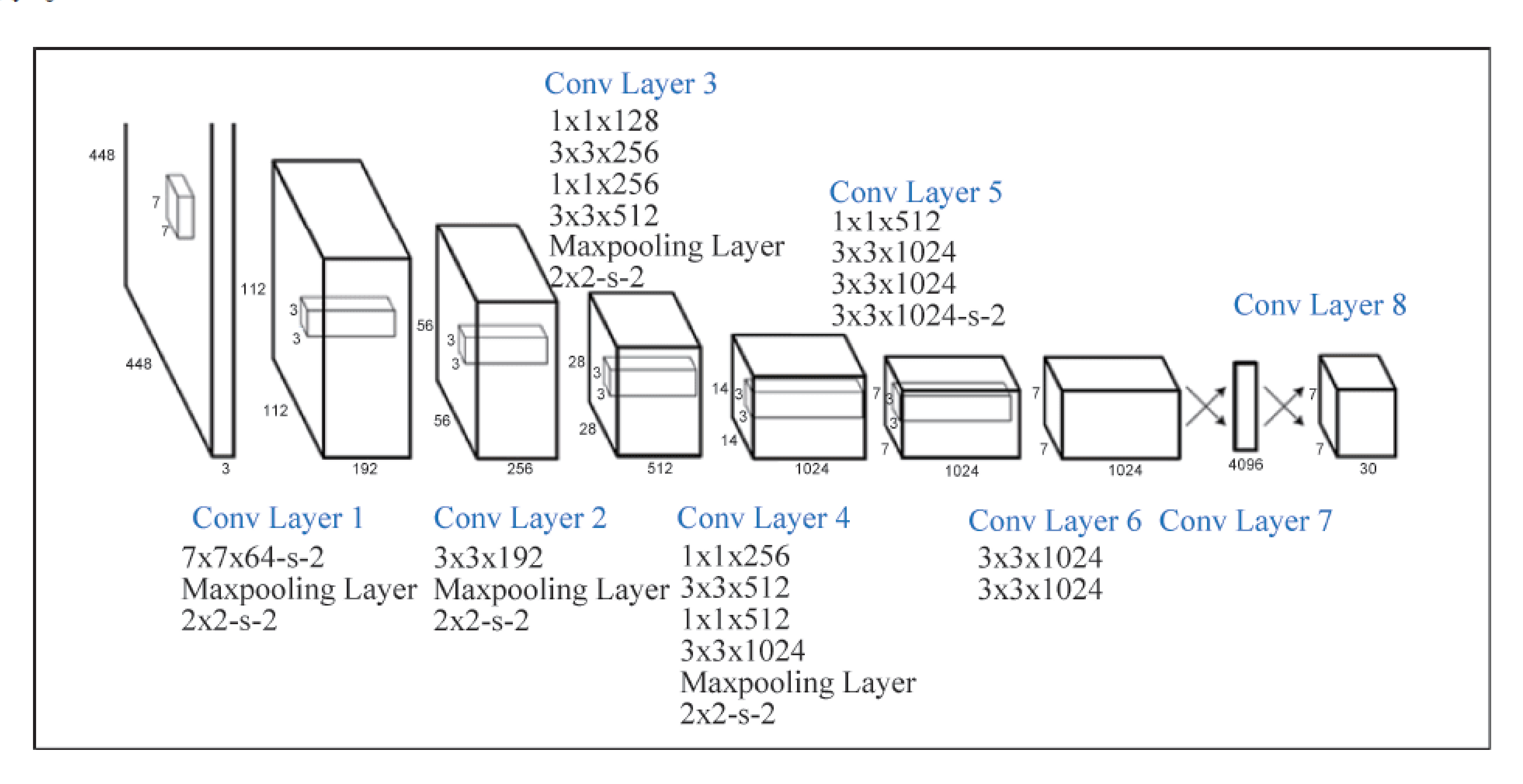
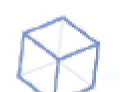


图 15.9　YOLO 中所用到的深度神经网络结构示意图

整个网络包含了 24 个 CNN 层，2 个全连接层，与 GoogleNet 不同的是：这里在用 3×3 卷积核得到的特征图后面用一个 1×1 的卷积操作来减少特征图的数量，例如，从 3×3×1024→1×1×512，这样可以减少计算的中间参数，减小模型的规模，从而加快运算速度。在模型的最后通过一个全连接层去预测各个图像块对应的目标类别和目标边界框，这样每一个图像块都用到了这个图片全部的特征，对于每个目标来说都考虑到了它存在的周围环境信息。

为了优化整个模型，作者设计了如下损失函数：

$$\begin{aligned}\text{loss} =& \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \cdot [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ &+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \cdot \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i}\right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i}\right)^2\right] \\ &+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{obj}} \cdot (C_i - \hat{C}_i)^2 + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{\text{noobj}} \cdot (C_i - \hat{C}_i)^2 \\ &+ \sum_{i=0}^{S^2} 1_i^{\text{obj}} \cdot \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \end{aligned} \tag{15.10}$$



参数 λ_{coord} 是用来增强边界框在损失计算中的重要性，$\lambda_{coord}=5$；参数 λ_{noobj} 是用来减弱非目标区域对目标区域置信度计算的影响，$\lambda_{noobj}=0.5$；1_i^{obj} 表示图像中第 i 个图像块有目标。1_{ij}^{obj} 表示第 i 个图像块的第 j 个预测框有目标，反之为 1_{ij}^{noobj}；式(15.10)的前两行是用来预测目标的边界框的，这里用平方根处理边界框宽和高 (w,h)，是因为在宽和高预测时，预测结果对大目标和小目标的影响是相同的，然而真实情况是 (w,h) 变化对大目标的敏感性要差于小目标，可能 (w,h) 变化一点，小目标的边框就无法找到，所以加入了这个开方过程。损失函数的第三项是用来预测边框置信度分数。第四项是预测非目标的置信度分数。最后一项是用来预测图像中每一个格子所属目标的类别。

整个深度学习模型首先用 ImageNet 1000 类的数据集来预训练，然后再用目标检测的数据集来微调整个网络并训练最后的预测器。这里用 VOC2007+VOC2012 数据集的训练集和验证集来共同训练整个网络。在训练时为了表述更细致的视觉信息，作者将深度网络输入图像大小调整为 224×224→448×448。训练参数有：批量大小为 64，动量项为 0.9，衰退参数为 0.0005，整个数据训练 135 次，第一次学习率从 0.001 变到 0.01，用 0.01 训练 75 次，0.001 训练 30 次，0.0001 训练最后 30 次。为了避免过拟合，模型中加了 dropout 策略，也做了数据增强。

训练结束后文中用了 VOC2007 和 VOC2012 的测试集测试了算法性能，结果展示于 15.3.3 小节。测试后将得到很多预测结果，这些预测结果通过非最大抑制(NMS)来筛选其中最佳的结果。

上述总结了 YOLO 算法的动机、优缺点，详细介绍了整个模型框架，并对模型进行了分析，概述了整个模型的训练和测试过程。

15.3.2　SSD

由于之前主流的目标检测和识别方式都是深度学习模型结合区域和高性能分类器共同完成检测识别任务。虽然这种方式能够得到较好的检测识别精度，但是整个模型的实现需要做大量的计算，要求高端的硬件，且无法做到实时处理。因此 Wei Liu 等人也提出了一种基于回归的目标检测与识别方法 SSD(Single Shot MultiBox Detector)，与 YOLO 类似，SSD 同样是一种端到端的模型，所有检测和识别过程都可以由同一个网络解决。与 YOLO 不同的是：SSD 在 YOLO 的基础上加入了 Faster R-CNN 中 anchor 机制，这样相当于在回归的基础上结合了一部分区域建议的功能，对比两者所用特征，SSD 并没有用这个图像的全局特征，只用了每个目标周围的深层特征去检测识别目标，而且在深度学习模型的特征提取上，SSD 从深度神经网络不同层的特征图上提取特征，然后分别用这些特征回归预测目标，这样能够自然地加入多尺度信息，能够对一个目标做更多的判断，从而在不影响速度的前提下提高精度。

SSD 算法的优点：①是一个 single-shot 目标检测识别方法，与 YOLO 类似，只需要观测图片一次就可以做到多目标的检测识别，速度比 YOLO 更快；②在 YOLO 的基础上结合



了 Faster R-CNN 中的 anchor 机制，而且用到了不同尺度的深度网络特征图预测每个位置上的目标，保证了在检测和识别的精度上可以和基于区域建议的方法相比；③SSD 算法对低分辨率的图像同样能达到较高的检测识别精度。

SSD 仍然没有解决目标的尺寸对检测结果的影响，对于一些小的目标做边界框的预测时没有一些基于区域建议的方法效果好，但 SSD 的优势暂时仍然是其他算法所不能比的。

SSD 算法的主要过程可以分为如下几个部分：①通过深度神经网络提取整个输入图片的深度特征；②针对不同尺度的深度特征图设计不同大小的特征抓取盒(将这些盒与真实目标边框相匹配用来训练)；③通过提取这些特征抓取盒对应的深度特征图的特征来预测盒中目标类别以及目标真实边框；④最终通过 NMS 来筛选最佳预测结果。

SSD 模型在训练时只需要图像、图像中目标类别和位置的真实标记，并不需要其他信息，在测试时也只需要输入一张没有经过处理的图片。整个模型可以用如图 15.10 所示。

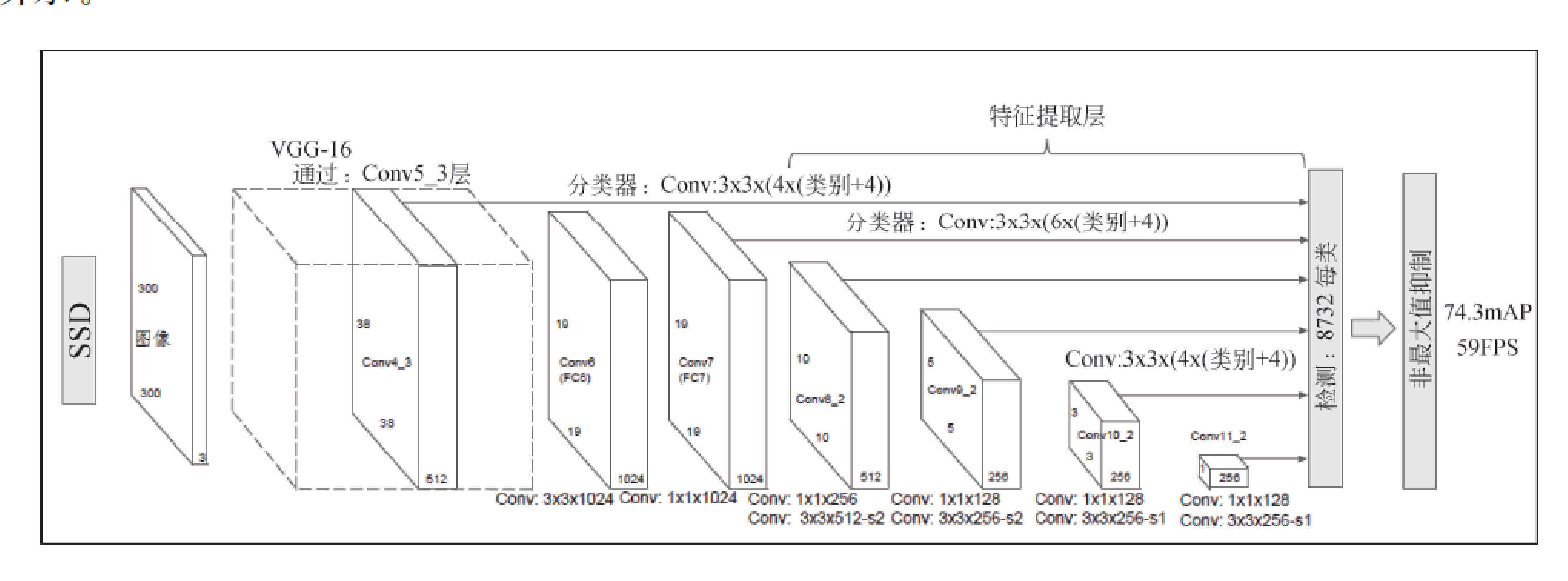
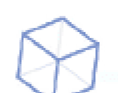


图 15.10　SSD 深度学习模型

可以看出整个模型的输入是整张图片，然后经过一个基础的深度学习模型 VGG16 网络来对整张图像提取特征，在 VGG16 网络的后面又加入了新的 CNN 层，由于每个 CNN 层的尺度是不一样的，这方便做多尺度特征的提取。后续用来检测识别的特征图包含 conv5_3、conv7、conv8_2、conv9_2、conv10_2、conv11_2。它们由如下方式产生：conv8_2→conv9_2，首先用 $3\times3\times256-s2$ 的卷积核来处理 conv8_2 层，其中 $s2$ 表示卷积步长是 2，这里的卷积过程用的是 atrous 算法，然后用 $1\times1\times128$ 的卷积核再做一次卷积计算，特征图数量从 256 变为 128，最终得到用于检测识别的特征图。接下来就是做多尺度特征图上的局部特征提取，并将得到的特征用于预测结果。对比于 YOLO 对原始图像划分的栅格，SSD 依次对特征图上的所有点做处理，比如特征图大小为 8×8 和 4×4，然后以每个特征图上对应坐标的特征向量预测目标。由于特征图是多尺度的，目标的大小也不是固定的，将使得这种简单地按照坐标点的特征提取方法无法很好地抓取目标。因此，Wei Liu 等人针对不同尺度的特征图、不同尺寸的目标，对每个特征图上的点重新设计了多重的特征抓取盒。对每个特征抓取盒可以用如下方式定义，首先对不同尺度的特征图设计不同的尺度参数，假定有 m 个特



征图，则尺度参数为：

$$s_k = s_{\min} + \left(\frac{s_{\max} - s_{\min}}{m-1}\right) \cdot (k-1), \quad k \in [1,m] \tag{15.11}$$

其中 $s_{\min}=0.2$ 为尺度参数最小值，$s_{\max}=0.9$ 为最大尺度参数。随后针对不同目标大小，目标长宽比设计特征抓取盒，先是 5 种不同长宽比参数 $a_r \in \left\{1,2,3,\frac{1}{2},\frac{1}{3}\right\}$，根据这组参数可以计算出特征抓取盒的宽和高，盒宽为 $w_k^a = s_k \sqrt{a_r}$，盒高为 $h_k^a = s_k / \sqrt{a_r}$，对于长宽比为 1 的盒，增加一组尺度参数 $s_k' = \sqrt{s_k s_{k+1}}$，则对于每一个特征图坐标点，可以得到 6 种不同的特征抓取盒。当然这些特征抓取盒并不是每一个都能对应一个目标，在训练阶段还需要筛选出这些盒里有效的特征抓取盒当作正样本训练。这里用到了 best jaccard overlap 来做特征抓取盒与真实目标位置的匹配。整个盒操作如图 15.11 所示。

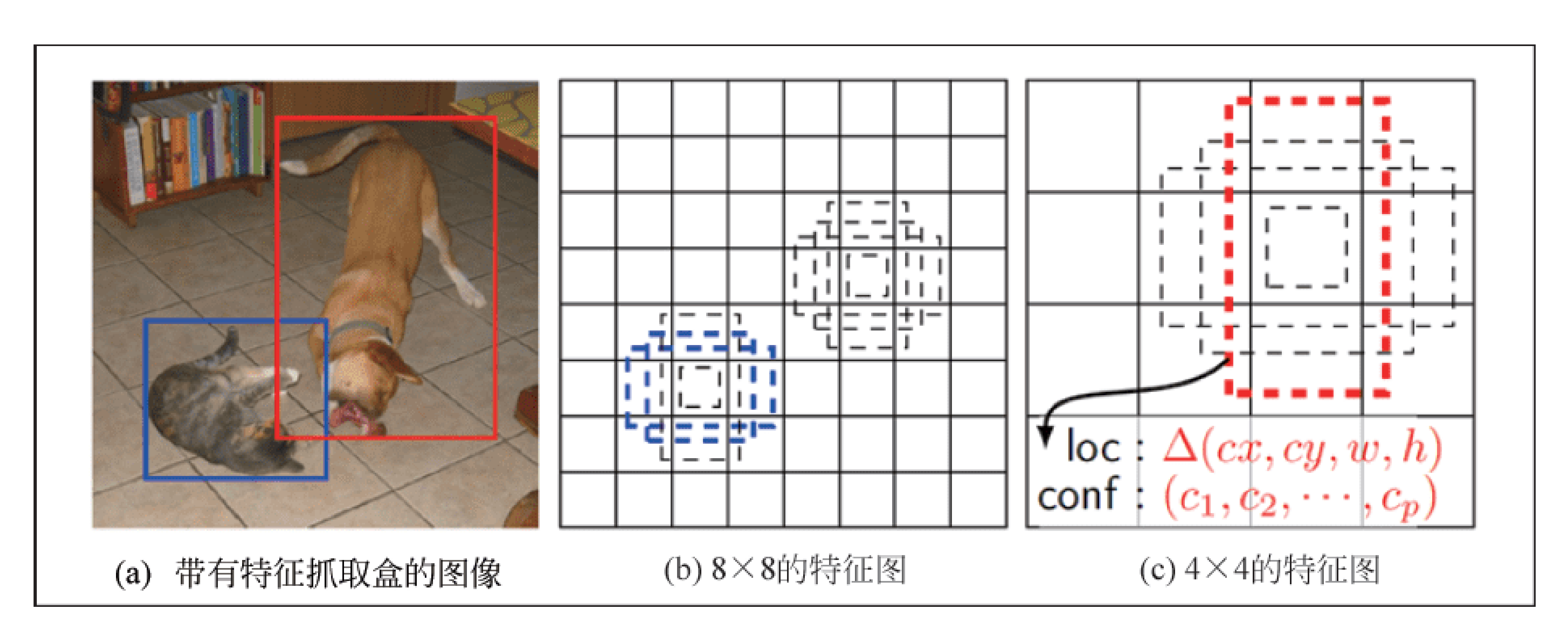


图 15.11　特征盒设计

接下来根据每个特征图坐标点上设计好的特征抓取盒来提取特征，并将这些特征用来预测目标的类别和边界框。这里用到了 3×3 的卷积核去提取每个特征抓取盒中的特征，每种特征图用到的卷积核为 3×3×6×(class+4)，6 为每个特征图坐标点上特征抓取盒的数量，若用 VOC 数据则 class=20，4 为预测的目标边界框与真实目标边界框之间的偏差。若一个特征图的尺寸为 $m \times n$，每个坐标上有 6 个盒，则最终产生 $m \times n \times 6 \times (\text{class}+4)$ 的输出结果。

整个模型的损失函数为：

$$L(x,c,l,g) = \frac{1}{N} \cdot (L_{\text{conf}}(x,c) + \alpha \cdot L_{\text{loc}}(x,l,g)) \tag{15.12}$$

其中 x 用来判断设计的特征抓取盒是否有对应的目标，$x_{ij}^p = \{1,0\}$ 表示第 i 个盒是否与第 p 类物体的第 j 个目标的边界框相匹配，匹配为 1，反之为 0。若 $\sum_i x_{ij}^p \geqslant 1$ 表示对于第 j 个目标边界框至少有一个盒与之匹配。式中 N 表示匹配盒的数量。式(15.12)中第一部分是用来衡量识别性能的，主要就是一个多类的 Softmax 损失函数，细节如下：

$$L_{\text{conf}}(x,c) = -\sum_{i\in \text{pos}}^{N} x_{ij}^{p}\log(\hat{c}_i^p) - \sum_{i\in \text{neg}}\log(\hat{c}_i^0) \tag{15.13}$$

其中 $\hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$。第二部分是用来衡量边界框预测性能的，用到的损失函数与 Fast R-CNN 相似，从预测的目标边框的偏差到目标边框的计算方式与 R-CNN 用到的方法相似，细节如下：

$$L_{\text{loc}}(x,l,g) = \sum_{i\in \text{pos}}^{N} \sum_{m\in\{cx,cy,w,h\}} x_{ij}^{k} \cdot \text{Smooth}_{L1}(l_i^m - \hat{g}_j^m) \tag{15.14}$$

其中$\hat{g}_j^m$ 表示第 j 个目标的真实目标边框与特征抓取盒的边框之间的偏差，$m\in\{cx,cy,w,h\}$，(cx,cy)表示边框中心点坐标，(w,h)表示边框的宽和高。

$$\hat{g}_j^{cx} = \frac{(g_j^{cx} - d_i^{cx})}{d_i^w} \tag{15.15}$$

$$\hat{g}_j^{cy} = \frac{(g_j^{cy} - d_i^{cy})}{d_i^h} \tag{15.16}$$

$$\hat{g}_j^{w} = \log\left(\frac{g_j^w}{d_i^w}\right) \tag{15.17}$$

$$\hat{g}_j^{h} = \log\left(\frac{g_j^h}{d_i^h}\right) \tag{15.18}$$

模型在训练时，先用预训练好的 VGG 模型，然后用 VOC2007 和 VOC2012 数据集中所有的训练和验证数据作为目标检测与识别模型的训练数据。训练过程中利用 SGD 策略优化目标函数，初始学习率为 0.001，动量项为 0.9，权重惩罚项为 0.0005，一次处理的图像批量为 32。

训练结束后，用测试数据来检验算法性能，结果展示于 15.3.3 节，在得到大量的预测结果后，通过 NMS 策略筛选结果，得到最佳的检测结果。

15.3.3 对比实验结果与分析

这里将基于回归的目标检测与识别算法和基于区域建议的检测识别算法做比较。整体来说在精度方面基于回归的方法仍有待提高，但是在算法效率方面，基于回归的方法有着较大的优势。通过对 VOC2012 数据测试来对比各种算法。从表 15.2 可以观察出，单纯的回归式检测算法 YOLO 在精度上无法与基于区域建议的方法比较。但是 SSD 在回归模型的基础上，通过加入一定候选区域筛选方法，结合了多尺度特征使得它达到了与基于区域建议的算法同级别的精度。但是这些算法仍然对一些小目标非常敏感，在比如 bottle 和 chair 等目标的检测上，精度仍然不能令人满意。

从算法效率角度来评价回归式检测算法，通过实验可以得出，YOLO 可以做到每秒处理 45 张图片，SSD 可以做到每秒处理 58 张图。远远超越了基于区域建议的目标检测算法。使得现有的目标检测算法可以做到实时检测。这是整个目标检测领域的又一突破性进展。

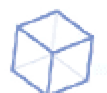


表 15.2　VOC2012 测试数据检测结果(%)

Detectors	R-CNN	Fast R-CNN	Faster R-CNN	YOLO	SSD
mAP	62.4	68.4	70.4	57.9	72.4
aero	79.6	82.3	84.9	77.0	85.6
bike	72.7	78.4	79.8	67.2	80.1
bird	61.9	70.8	74.3	57.7	70.5
boat	41.2	52.3	53.9	38.3	57.6
bottle	41.9	38.7	49.8	22.7	46.2
bus	65.9	77.8	77.5	68.3	79.4
car	66.4	71.6	75.9	55.9	76.1
cat	84.6	89.3	88.5	81.4	89.2
chair	38.5	44.2	45.6	36.2	53.0
cow	67.2	73.0	45.6	60.8	77.0
table	46.7	55.0	55.3	48.5	60.8
dog	82.0	87.5	86.9	77.2	87.0
horse	74.8	80.5	81.7	72.3	83.1
mbike	76.0	80.8	80.9	71.3	82.3
person	65.2	72.0	79.6	63.5	79.4
plant	35.6	35.1	40.1	28.9	45.9
sheep	65.4	68.3	72.6	52.2	75.9
sofa	54.2	65.7	60.9	54.8	69.5
train	67.4	80.4	81.2	73.9	81.9
tv	60.3	64.2	61.5	50.8	67.5

15.4　基于学习搜索的目标检测与识别

前文讲述的主流的目标检测与识别算法，不论是基于区域建议或者是基于回归的方法，都是直接对图像或者图像的深度特征图划定区域，然后针对各个区域得到检测识别结果。而现如今也存在一些其他方式来处理目标检测与识别任务的方法。例如说，Juan C. Caicedo 等人和 Míriam Bellver Bueno 等人提出用深度强化学习方式在图像中搜索目标并识别搜索到的目标，而 Donggeun Yoo 等人提出用视觉注意结合深度学习的方式来搜索识别图像中的目标。后续将详细介绍 Juan C. Caicedo 和 Donggeun Yoo 的两种搜索方式。

15.4.1　基于深度学习的主动目标定位

主流的基于深度学习的目标检测与识别算法是通过判断一些候选区域来达到检测和识别目标的目的的。而作者认为可以将目标定位这一过程看作一个通过一些动作不断调整边框从而达到搜索目标的任务，也可以看作一个动态视觉搜索任务。作者通过深度强化学习

的方式来预测每一次边框变动的动作，再根据执行动作的结果来判断目标是否找到。

这种主动搜索的方法与大多数方法都不同，与滑窗搜索的检测识别方法对比，这种方法对不同目标、不同场景中有着不同的搜索步长。与区域建议的检测方式相比，这种方法是通过深度强化学习的方式来搜索最可能的目标候选区域，而不是通过一些简单的、非学习的方式来枚举目标候选区域。在目标边界回归上，这种方法不会通过单次的、固定的方式去预测目标。

就目前来说，这种方法还无法超越基于区域建议的目标检测识别方法，主要有如下缺陷：

- 由于要在图像上主动搜索目标，每找一个目标就需要多次搜索，因此在效率上无法超越所有现有的目标检测识别方法。
- 在搜索过程中容易出现定位到目标但 Iou 指数不满足要求的情况，这使得这种方法在精度上无法与主流目标检测与识别方法相比较。
- 算法对目标大小比较敏感。
- 当同一场景中有多个同类目标或者目标之间有遮挡时，也对检测结果有较大影响。

整个模型首先通过深度神经网络对图像提取特征，将这个特征与前几次预测的动作向量级联起来组成新的特征向量，然后将这个组合特征向量送入深度 Q 学习网络来预测图像下一步的移动方式，通过奖励机制来判断移动是否有效。当模型预测动作停止后对提取的图像块做目标类别的识别。整个模型可以用图 15.12 来表示。

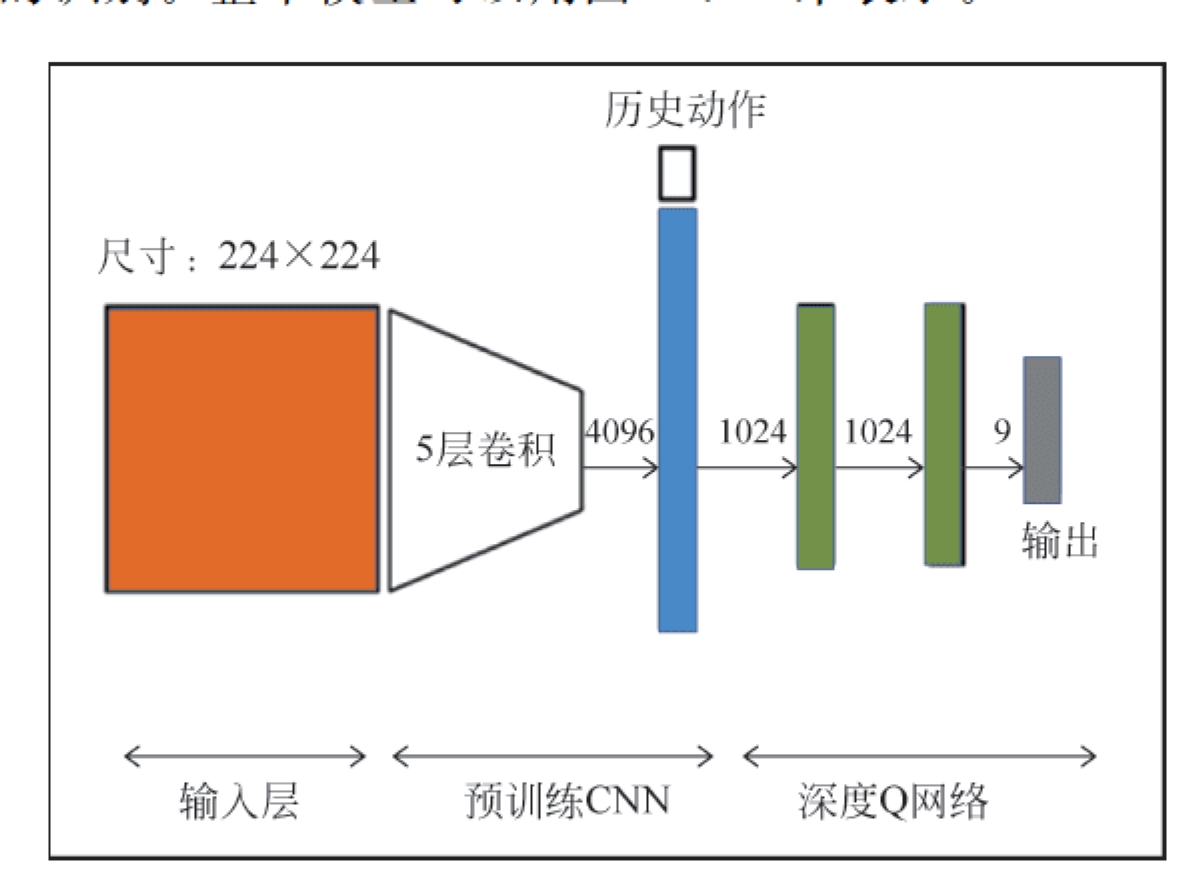


图 15.12　基于深度 Q 学习的目标检测识别模型

整个模型输入为固定的 224×224 的图片，将这个图片送入有 5 个卷积层和 1 个全连接层的深度特征提取网络，这样就得到一个 4096 维的特征向量，这个特征向量中包含了图像的全局信息。作为一个序列的动作预测过程，这里还结合了当前状态之前的动作集合，这些动作集合组成的向量与图像的特征向量组成了当前区域的状态描述，(o,h)其中 o 表示图像的特征，h 表示历史动作序列，共 10 组历史动作，状态向量维度为 4096＋90。为了表示搜索

过程，作者定义了 9 种动作，包括左右平移、上下平移、尺度放大或缩小、缩短高或者缩短宽、最后是动作停止，如图 15.13 所示。

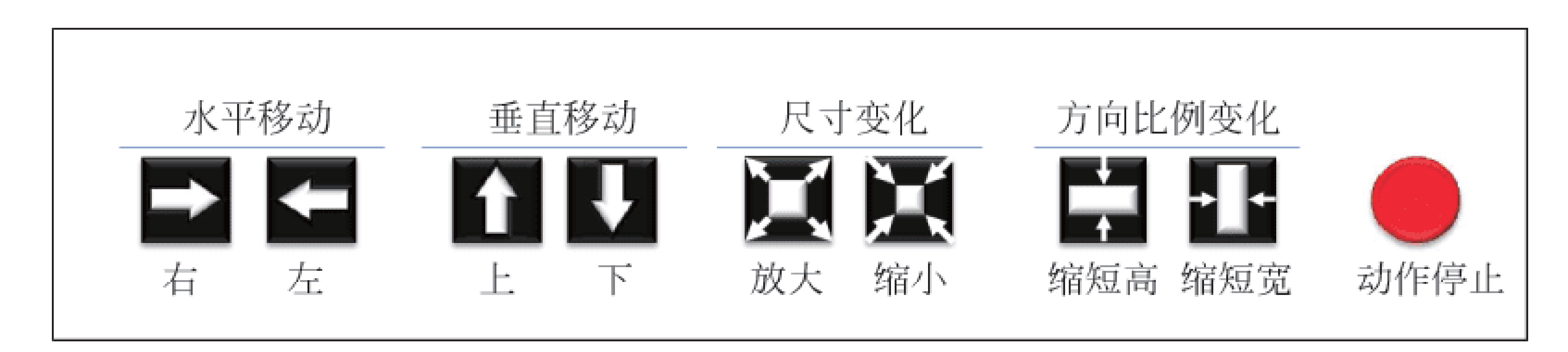


图 15.13　搜索中的 9 种动作

得到状态向量后，将状态向量送入深度 Q 网络来预测下一步的动作向量。每次变动的幅度由式(15.19)来表示：

$$\begin{cases} \alpha_w = \alpha \cdot (x_2 - x_1) \\ \alpha_h = \alpha \cdot (y_2 - y_1) \end{cases} \tag{15.19}$$

其中 α_w 是水平方向上的变化参数，α_h 是垂直方向上的变化参数，α 是限制参数，这里为 0.2。结合预测到的下一次的动作以及动作的变化幅度在输入图像上截取下一次用到的输入图像。

Juan C. Caicedo 等人设计了奖励函数来评价每一次动作的有效性。奖励函数可以解释为当根据一个动作从状态 s 变动到 s'，通过两个状态所在区域与真实目标区域之间的 IOU 指数来判断动作有效性，当有效时给予奖励，反之惩罚。具体可以用式(15.20)来表示：

$$R_a(s,s') = \text{sign}(\text{Iou}(b',g) - \text{Iou}(b,g)) \tag{15.20}$$

g 为真实目标边框，b 为当前状态区域边框，b' 为执行动作之后的下一状态的区域边框。当预测到终止状态时，将用另一种奖励函数：

$$R_w(s,s') = \begin{cases} +\eta & \text{Iou}(b,g) \geq \tau \\ -\eta & \text{其他} \end{cases} \tag{15.21}$$

这里 $\tau=0.6$，$\eta=3$。可以预计：当模型一直向着目标的位置去移动时，奖励分数是不断累积的，只要方向正确，分数会越来越大。用最后累加的奖励分数来评价整个搜索过程的动作序列。最终动作选择的时候可以用式(15.22)来选取具有最大奖励分数的动作序列：

$$Q(s,a) = r + \gamma \cdot \max_a Q(s',a') \tag{15.22}$$

其中 $Q(s,a)$ 表示状态 s 情况下，执行动作 a 后所能得到的最大奖励分数，r 表示当前动作的奖励分数，$\max_a Q(s',a')$ 表示执行动作 a 后所能达到的最大奖励分数。根据这些动作序列就可以找到对应的目标。但是这种动态搜索的效果仍然不能和基于区域建议的深度学习相比，但是这种方法不需要产生大量的目标候选区域，不需要做大量的重复计算，在这些方面这种方法有一定的优势。然而随着 Faster R-CNN 等高效的基于区域建议的方法，以及 SSD 等回归模式的目标检测方法的相继出现，这种方法暂时还没有什么太大的优势，但是这种动态搜索的目标检测与识别方法仍然是一种比较有效的方式，仍有很大的提升空间。



15.4.2 AttentionNet

对比于 Juan C. Caicedo 等人通过深度强化学习的方式来搜索目标，Donggeun Yoo 等人设计了一个直接基于 CNN 的 AttentionNet 模型来预测目标窗的行动过程，这里将动作预测当作一个分类任务来处理，没有用到深度强化学习的方式。这样节省了深度 Q 学习网络，相当于在相同基本深度学习模型下，AttentionNet 模型规模会小一些，而且也不需要前期的动作序列作为附加特征来判断将要移动的动作。

AttentionNet 模型可以概述为：先取一个足够大能包含整个目标的框，提取框内图像，将图像送入深度神经网络提取特征，用这组特征来预测框的移动动作，判断框是否有目标，没有目标将判断这个框为负样本，有目标若满足条件则表示检测完成，若没有满足条件则根据预测动作提取新的图像作为下一次检测的输入，不断重复这个过程直到检测结束。整个模型可以用图 15.14 来表示。

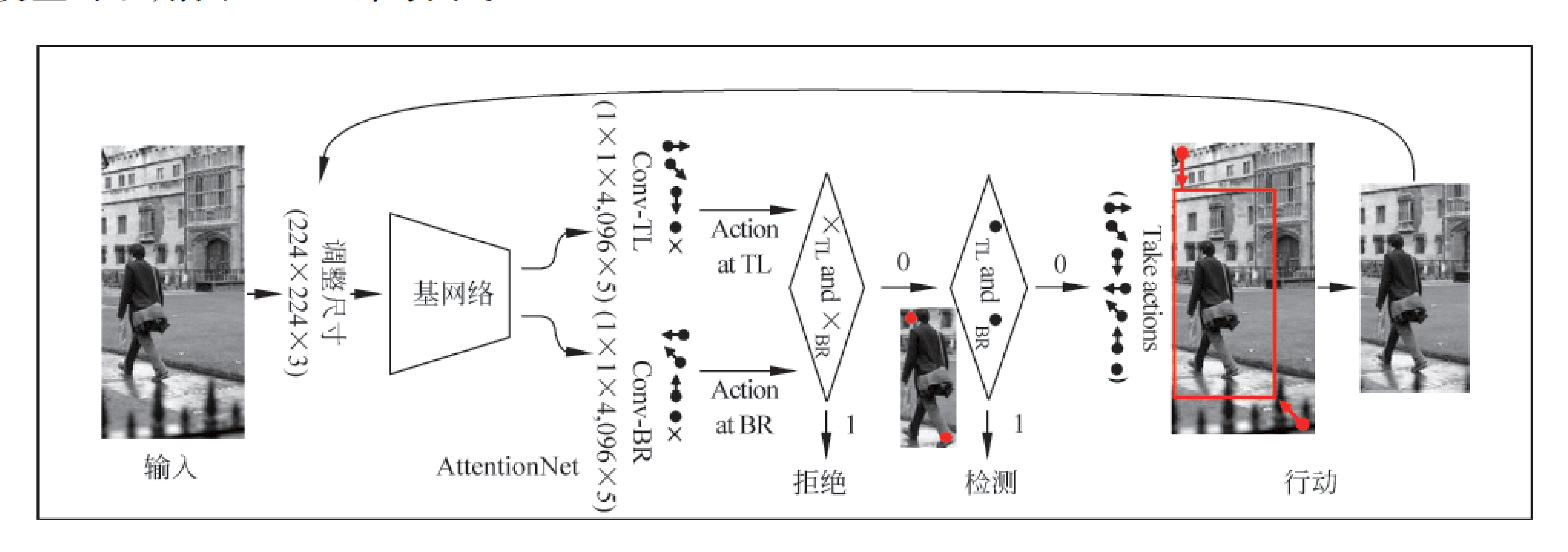


图 15.14　AttentionNet 模型

在训练时先要找到包含目标的区域作为初始视觉注意点，这里作者在以真实目标边框为中心缩小到原来的$\frac{1}{2}$的边框和扩大 6 倍的边框区域之间随机选取一定数量的边框作为训练初始正样本边框。选取一些与真实目标边框不重叠的边框作为负样本。为了保证正样本的边框中包含目标，先用边框内的图像预测边框下一步的移动动作，若动作为右下移↘和左上移↖，则说明框内包含完整的目标，将这样的边框称为初始视觉注意点。

选好目标框后抓取框内图像，并将图像大小调整成固定的 224×224 大小。然后将图像送入基本深度神经网络模型做特征提取，在网络的最后并没有用全连接层来做预测，而是通过不同大小的卷积核用卷积操作来得到最后的动作向量。不同于上一小节方法的目标框动作，这里设计的动作只针对边界框的左上角点和右下角点的移动。左上角点的动作包含右移→、下移↓、右下移↘、无目标×、移动终止•。右下角点的动作包含左移←、上移↑、左上移↖、无目标×、移动终止•。由于有两个动作基点，因此在网络最后将输出两组动作预测结果，一组表示左上角点，一组表示右下角点。整个模型的损失函数可以定义为：

$$l=\frac{1}{2}\cdot l_{\text{softmax}}(\boldsymbol{y}_{\text{TL}},\boldsymbol{t}_{\text{TL}})+\frac{1}{2}\cdot l_{\text{softmax}}(\boldsymbol{y}_{\text{BR}},\boldsymbol{t}_{\text{BR}})$$

$$\text{s.t.}\quad l_{\text{softmax}}(\boldsymbol{y},\boldsymbol{t})=-y_t+\log\sum_i e^{y_i} \tag{15.23}$$

其中 $\boldsymbol{y}$ 和 $\boldsymbol{t}$ 为 5 维的动作向量，$\boldsymbol{y}$ 表示预测的动作向量，$\boldsymbol{t}$ 为真实动作的标签。若做多类的检测时，这个损失函数将定义为：

$$l=\lambda\cdot l_{\text{cls}}+\frac{1-\lambda}{2}\cdot(l_{\text{TL}}+l_{\text{BR}})$$

$$\text{s.t.}\quad l_{\text{cls}}=l_{\text{softmax}}(\boldsymbol{y}_{\text{cls}},\boldsymbol{t}_{\text{cls}})$$

$$\begin{cases} l_{\text{TL}}=\sum_{c=1}^{N}1(c,\boldsymbol{t}_{\text{cls}})\cdot l_{\text{softmax}}(\boldsymbol{y}_{\text{TL}c},\boldsymbol{t}_{\text{TL}}) \\ l_{\text{BR}}=\sum_{c=1}^{N}1(c,\boldsymbol{t}_{\text{cls}})\cdot l_{\text{softmax}}(\boldsymbol{y}_{\text{BR}c},\boldsymbol{t}_{\text{BR}}) \\ l_{\text{softmax}}(\boldsymbol{y},\boldsymbol{t})=-\boldsymbol{y}_t+\log\sum_i e^{\boldsymbol{y}_i} \end{cases} \tag{15.24}$$

当得到网络预测的一组动作后，先判断是否满足终止条件。若左上角点和右下角点预测的动作都为×则表示这个区域是没有目标的，网络将不再对这个区域做多余的判断，并检测其他边框。当两组预测动作都为·时，表示已找到满足条件的目标区域，并且不会再做边框的移动了。其他情况下则按照对应动作来移动边框，将边框内的图片取出作为下一次模型的输入。

当做多类的目标检测与识别时，整个深度网络模型每次对一个边界框做多类的移动动作预测，并预测边框内目标类别。这里不需要预测无目标×的情况，将这种情况放在类别预测步骤中，将其定为背景类。这样当检测到一个目标的同时预测了目标的类别。剩下的步骤与单类目标检测相同，整个模型可以用图 15.15 来表示。对于基于区域建议的目标检测与识别方法(R-CNN 等)是用每一个候选区域中目标的类别概率作为检测评分。而在 AttentionNet 中，作者通过没有经过 Softmax 的预测的动作向量值来计算每一个边界框的检测分数。可以用式(15.25)来表示。

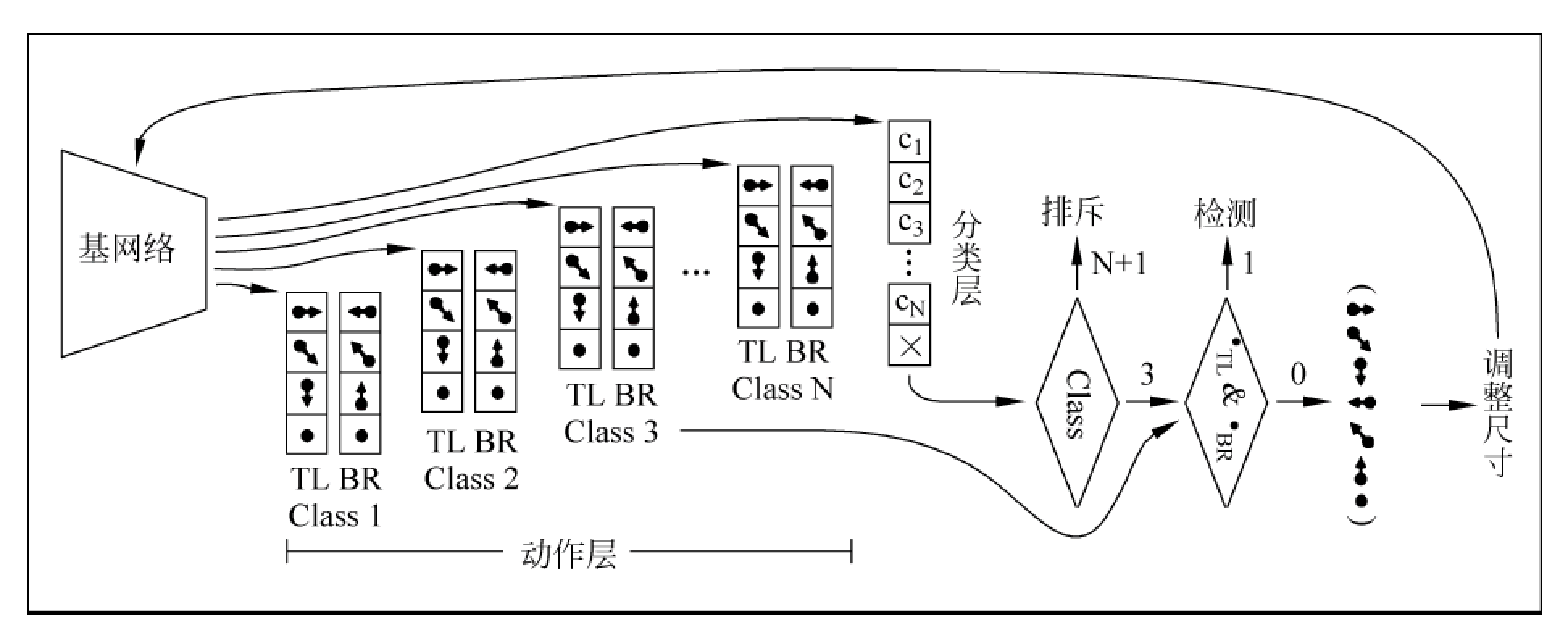


图 15.15　多类目标检测的 AttentionNet 模型

$$\boldsymbol{s}^{b}=\frac{1}{2}(\boldsymbol{s}_{\mathrm{TL}}^{b}+\boldsymbol{s}_{\mathrm{BR}}^{b})$$

$$\text{s. t.}\begin{cases}\boldsymbol{s}_{\mathrm{TL}}^{b}=\boldsymbol{y}_{\mathrm{TL}}^{\cdot}-(\boldsymbol{y}_{\mathrm{TL}}^{\rightarrow}+\boldsymbol{y}_{\mathrm{TL}}+\boldsymbol{y}_{\mathrm{TL}}^{\downarrow}+\boldsymbol{y}_{\mathrm{TL}}^{\times})\\ \boldsymbol{s}_{\mathrm{BR}}^{b}=\boldsymbol{y}_{\mathrm{BR}}^{\cdot}-(\boldsymbol{y}_{\mathrm{BR}}^{\leftarrow}+\boldsymbol{y}_{\mathrm{BR}}+\boldsymbol{y}_{\mathrm{BR}}^{\uparrow}+\boldsymbol{y}_{\mathrm{BR}}^{\times})\end{cases}\tag{15.25}$$

在做多类检测时，则不能只用动作向量来评价检测结果，还需要将对类别的预测结果加入评价参数中。可以用式(15.26)来表示：

$$\boldsymbol{s}^{b}=(1-\lambda)\boldsymbol{s}_{\mathrm{cls}}^{b}+\lambda(\boldsymbol{s}_{\mathrm{TL}}^{b}+\boldsymbol{s}_{\mathrm{BR}}^{b})$$

$$\text{s. t.}\quad \boldsymbol{s}_{\mathrm{cls}}^{b}=\boldsymbol{y}_{\hat{c}}-\boldsymbol{y}_{N+1}$$

$$\begin{cases}\boldsymbol{s}_{\mathrm{TL}}^{b}=\boldsymbol{y}_{\mathrm{TL}}^{\cdot}-(\boldsymbol{y}_{\mathrm{TL}\hat{c}}^{\rightarrow}+\boldsymbol{y}_{\mathrm{TL}\hat{c}}+\boldsymbol{y}_{\mathrm{TL}\hat{c}}^{\downarrow}+\boldsymbol{y}_{\mathrm{TL}\hat{c}}^{\times})\\ \boldsymbol{s}_{\mathrm{BR}}^{b}=\boldsymbol{y}_{\mathrm{BR}}^{\cdot}-(\boldsymbol{y}_{\mathrm{BR}\hat{c}}^{\leftarrow}+\boldsymbol{y}_{\mathrm{BR}\hat{c}}+\boldsymbol{y}_{\mathrm{BR}\hat{c}}^{\uparrow}+\boldsymbol{y}_{\mathrm{BR}\hat{c}}^{\times})\end{cases}\tag{15.26}$$

AttentionNet 的检测结果虽然高于 Juan C. Caicedo 的方法，但是相较于一些基于建议区域的检测算法或者回归的方法来说检测精度仍有待提高。

15.4.3 对比实验结果与分析

这里将基于搜索的目标检测与识别算法(Q-learning，AttentionNet)和其他检测识别算法做了比较。整体来说，在精度方面基于搜索的方法并不占有优势，但是基于搜索的方法仍然是一种有效的目标检测识别算法。通过对 VOC2007 数据测试来对比各种算法(见表 5.3)。

表 15.3 VOC2007 测试数据检测结果(%)

Detectors	DPM	R-CNN	Fast R-CNN	Faster R-CNN	SSD	Q-learning	AttentionNet
mAP	33.7	66.0	70.0	73.2	68.0	46.1	70.7
aero	33.2	73.4	77.0	76.5	73.4	55.5	79.1
bike	60.3	77.0	78.1	79.0	77.5	61.9	77.6
bird	10.2	63.4	69.3	70.9	64.1	38.4	70.2
boat	16.1	45.4	59.4	65.5	59.0	36.5	58.0
bottle	27.3	44.6	38.3	52.1	38.9	21.4	60.0
bus	54.3	75.1	81.6	83.1	75.2	56.5	75.8
car	58.2	78.1	78.6	84.7	80.8	58.8	85.5
cat	23.0	79.8	86.7	86.4	78.5	55.9	75.9
chair	20.0	40.5	42.8	52.0	46.0	21.4	47.6
cow	24.1	73.7	78.8	81.9	67.8	40.4	79.9
table	26.7	62.2	68.9	65.7	69.2	46.3	61.6
dog	12.7	79.4	84.7	84.8	76.6	54.2	76.9
horse	58.1	78.1	82.0	84.6	82.1	56.9	78.6
mbike	48.2	73.1	76.6	77.5	77.0	55.9	76.0
person	43.2	64.2	69.9	76.7	72.5	45.7	80.1
plant	12.0	35.6	31.8	38.8	41.2	21.1	47.0
sheep	21.1	66.8	70.1	73.6	64.2	47.1	73.9
sofa	36.1	67.2	74.8	73.9	69.1	41.5	64.3
train	46.0	70.4	80.4	83.0	78.0	54.7	74.1
tv	43.5	71.1	70.4	72.6	68.5	51.4	72.5

从表 15.3 中可以看出，基于 Q-learning 的方法在精度方面无法令人满意，基于 AttentionNet 的方法在精度方面已经接近其他优秀方法，但这些方法仍然对目标尺寸非常敏感，当目标尺寸较小或者形态变化多种多样时效果不太理想。

参考文献

[15.1] Forsyth D. Object Detection with Discriminatively Trained Part-Based Models[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2010, 32(9): 1627-1645.

[15.2] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]. International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012: 1097-1105.

[15.3] Girshick R, Donahue J, Darrell T, et al. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation[C]. Computer Vision and Pattern Recognition. IEEE, 2013: 580-587.

[15.4] Girshick R. Fast R-CNN[J]. Computer Science, 2015.

[15.5] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2015: 1-1.

[15.6] Redmon J, Divvala S, Girshick R, et al. You Only Look Once: Unified, Real-Time Object Detection[J]. 2015: 779-788.

[15.7] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector[J]. 2015.

[15.8] Yoo D, Park S, Paeng K, et al. Action-Driven Object Detection with Top-Down Visual Attentions[J]. 2016.

[15.9] Caicedo J C, Lazebnik S. Active Object Localization with Deep Reinforcement Learning[C]. IEEE International Conference on Computer Vision. IEEE, 2015: 2488-2496.

[15.10] Uijlings J R R, Sande K E A V D. Selective Search for Object Recognition[J]. International Journal of Computer Vision, 2013, 104(2): 154-171.

[15.11] Cheng M M, Zhang Z, Lin W Y, et al. BING: Binarized Normed Gradients for Objectness Estimation at 300fps[C]. Computer Vision and Pattern Recognition. IEEE, 2014: 3286-3293.

[15.12] Zitnick C L, Dollár P. Edge Boxes: Locating Object Proposals from Edges[M]. Computer Vision-ECCV 2014. Springer International Publishing, 2014: 391-405.

[15.13] Zhang Y, Sohn K, Villegas R, et al. Improving object detection with deep convolutional networks via Bayesian optimization and structured prediction[C]. Computer Vision and Pattern Recognition. IEEE, 2015: 249-258.

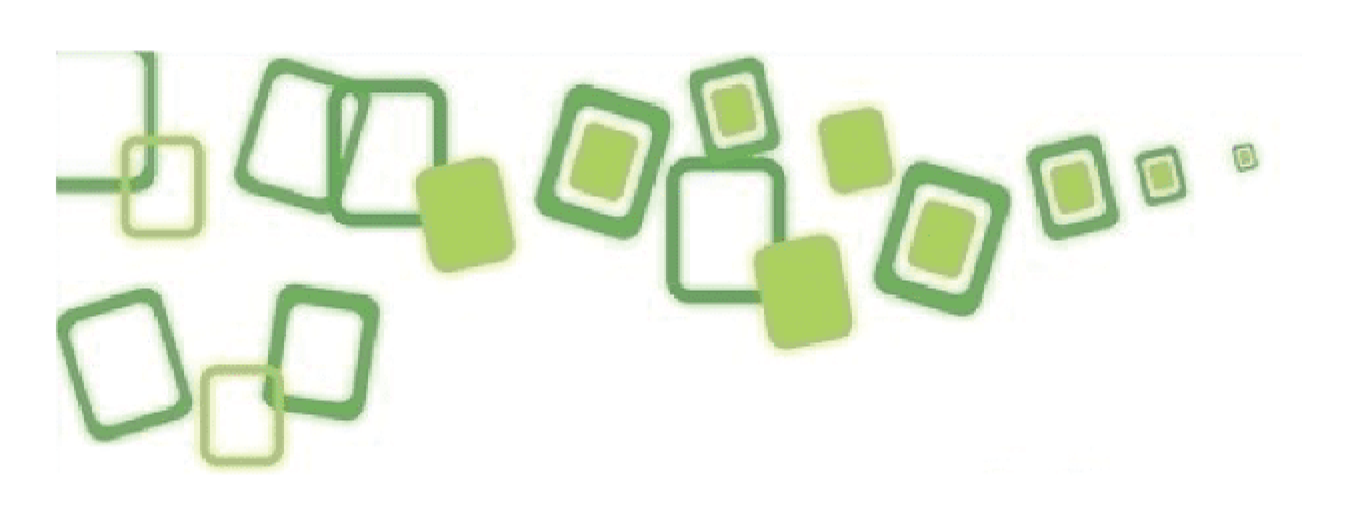
[15.14] Gidaris S, Komodakis N. Object Detection via a Multi-region and Semantic Segmentation-Aware CNN Model[C]. IEEE International Conference on Computer Vision. IEEE, 2015: 1134-1142.

[15.15] Zhu Y, Urtasun R, Salakhutdinov R, et al. segDeepM: Exploiting Segmentation and Context in Deep Neural Networks for Object Detection[J]. 2015, 84(84): 4703-4711.

[15.16] Kong T, Yao A, Chen Y, et al. HyperNet: Towards Accurate Region Proposal Generation and Joint Object Detection[J]. 2016: 845-853.

[15.17] Ouyang W, Luo P, Zeng X, et al. DeepID-Net: multi-stage and deformable deep convolutional neural networks for object detection[J]. e-Print arXiv, 2014.
[15.18] Bell S, Zitnick C L, Bala K, et al. Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks[J]. 2015: 2874-2883.
[15.19] Zhang N, Donahue J, Girshick R, et al. Part-based R-CNNs for Fine-grained Category Detection [J]. 2014, 8689: 834-849.
[15.20] Dai J, Li Y, He K, et al. R-FCN: Object Detection via Region-based Fully Convolutional Networks[J]. 2016.
[15.21] Bellver M, Giroinieto X, Marques F, et al. Hierarchical Object Detection with Deep Reinforcement Learning[J]. 2016.
[15.22] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science, 2014.
[15.23] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[J]. 2015: 1-9.
[15.24] Hinton G E, Srivastava N, Krizhevsky A, et al. Improving neural networks by preventing co-adaptation of feature detectors[J]. Computer Science, 2012, 3(4): págs. 212-223.
[15.25] Holschneider M, Kronland-Martinet R, Morlet J, et al. A Real-Time Algorithm for Signal Analysis with the Help of the Wavelet Transform[M]. Wavelets. Springer Berlin Heidelberg, 1990: 286-297.
[15.26] Erhan D, Szegedy C, Toshev A, et al. Scalable Object Detection Using Deep Neural Networks [J]. 2013: 2155-2162.
[15.27] Huval B, Coates A, Ng A. Deep learning for class-generic object detection[J]. Computer Science, 2013.
[15.28] Makantasis K, Karantzalos K, Doulamis A, et al. Deep Learning-Based Man-Made Object Detection from Hyperspectral Data[J]. 2015, 9474(1): 717-727.

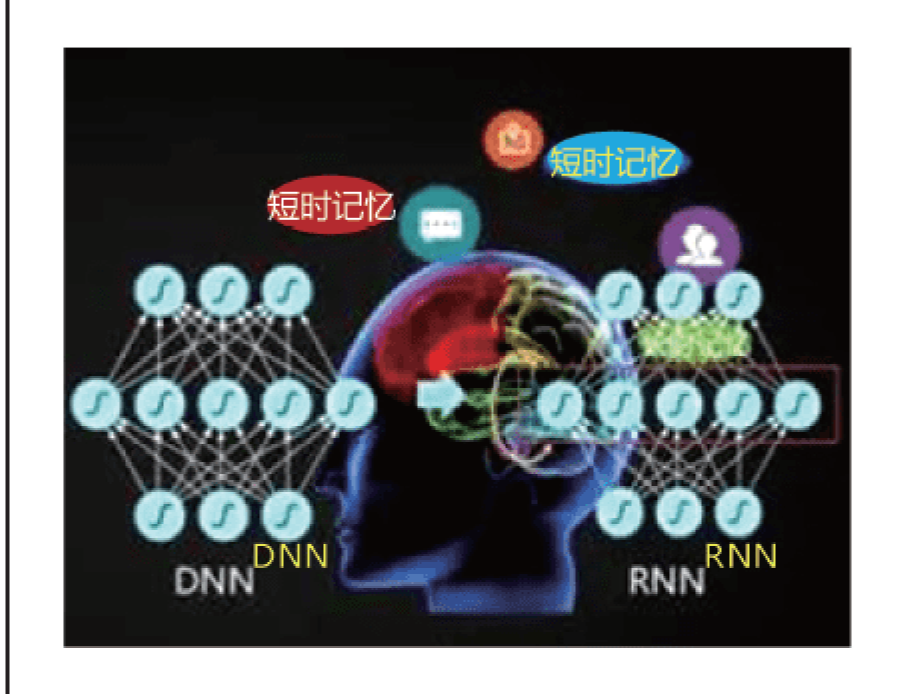
第16章



总结与展望

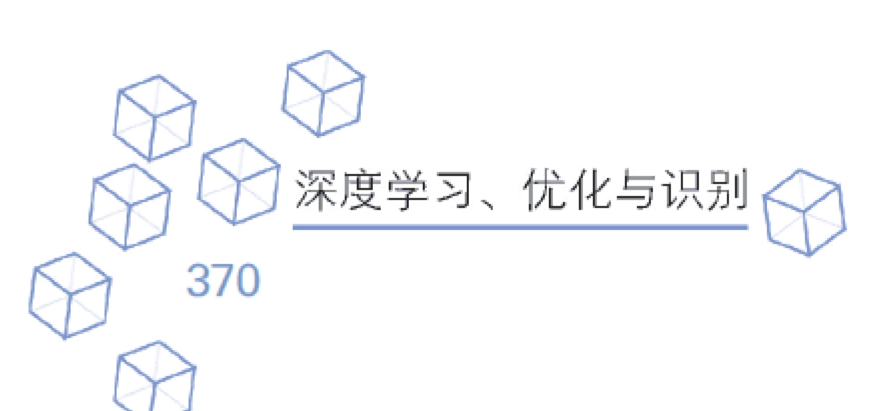
CHAPTER 16

深度学习促进了人工智能技术的革新





大数据·深度学习使得人工智能在诸多领域获得了突破性的进展

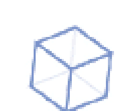


16.1 深度学习发展历史图

深度学习是人工智能领域最能体现智能的一个分支，它也是机器学习领域中的核心话题，研究深度学习的发展脉络（从 1958 年的感知机神经网络、到 20 世纪 70 年代人工智能的寒冬，到 1986 年反向传播算法（深度学习的关键性基础），再到 2006 年的深度置信网络），需要明白在大数据时代的背景下，硬件的发展带来计算能力的大幅度提升，促使"复杂结构"下的深度学习训练变得更为有效，另外数据的大幅度增加降低了过拟合风险，同时深层的表达方式也弱化了对数据先验知识的依赖性。下面我们将从以下两个方面来展开对于深度学习脉络的介绍，即 16.1.1 和 16.1.2 节；并在 16.1.3 节给出近十余年出现的各种深度神经网络名词的关系谱图。

16.1.1 从机器学习、稀疏表示学习到深度学习

对于分类问题，机器学习、稀疏表示学习与深度学习拥有着各自的处理技巧与策略，如机器学习中的支撑向量机，包括线性、半线性（数据到特征空间的转化，即特征学习）和非线性（核技巧）方式下的模型设计；稀疏表示学习中的稀疏表示分类器，包括基于字典学习的稀疏表示分类器、结构化稀疏表示分类器等；深度学习中的深度卷积神经网络、深度置信网络（整体结构包括特征学习、分类器设计两个子阶段）等。通常，机器学习的可释性数学框架包括数据、模型、优化和求解四个部分，其中的模型是核心（基于不同数据所作出的假设得到的模型也是不同的）；下面我们便从这四个方面陈述三者之间的关系，首先，对于数据的要求，机器学习和稀疏表示学习都是中小规模的，深度学习是中大规模的，并且机器学习注重挖掘数据中的先验知识并将其转化为相应的正则化约束或者人工设计特征（例如小波特征、Sift 特征，统计特征等），而稀疏表示学习则注重挖掘数据在某一框架下（如不同类数据所构造的或学习的字典）的稀疏特性和结构拓扑特性（如低秩约束所带来的全局拓扑特性的保持），深度学习则对数据的先验知识的依赖性较弱，期望通过层次表达的堆叠，学习到数据的内在规律特性，使得这种特性呈可分性或者判别特性等，尽可能弱化对分类器的复杂性设计，达到简单、新颖和通用的目的。其次，对于模型，依据数据的假设建立的机器学习和稀疏表示学习是较为简单的网络结构，即网络模型的复杂度低但内蕴较大（约束较多，可应用的场景受限），这意味着完成的学习任务较为简单且表征数据的能力有限；而深度学习通常建立的模型复杂度较高，而内蕴较小（问题的可应用范围广，即外延大），但模型的学习效率降低（参数量大，易出现过拟合现象），为了缓解这一问题，通常采用的策略是逐层学习（参数初始化的选取方式）和大幅度增加数据量。最后，对于优化和求解，机器学习和深度学习的关键性基础是反向传播算法，即目标函数通过利用链式法则，对于某层输入的导数（或者梯度）可以通过反向传播对该层输出（或者下一层输入）的导数求得，进而调整权值，不过，机器学习通常处理的任务是凸优化的，而深度学习则是高度非凸优化的；而稀疏表示学习则通过



特定的追踪匹配算法(OMP)或交替方向乘子 ADMM 算法实现目标函数的求解,包括凸优化问题(即 L_1 范数约束的 BP 算法等)和非凸优化问题(即 L_0 范数约束的 OMP 算法等)。

近年来,基于机器学习、稀疏表示学习和深度学习的方法相互结合,各自发挥其相应模块的优势,形成各种策略或特性下的深度神经网络模型,应用于各种任务。例如深度 PCA 网络,即深度卷积神经网络的参数由 PCA 算法进行初始化获取;稀疏深度神经网络,即深度神经网络中通过使用修正线性单元作为激活函数(内蕴稀疏特性),或使用 Dropout 策略(全连接变为稀疏连接),或使用池化操作(通过缩小尺寸或原尺寸补零操作,强调平移不变性),或将隐层节点稀疏性约束等引入至网络模型中,模拟生物视觉神经的稀疏响应特性,以及使用标准稀疏编码或稀疏表示进行字典学习,将得到的字典进行转置得到滤波器组作为深度神经网络的参数初始化选取的方式;深度小波/多尺度几何神经网络,仍采用深度神经网络的架构,将其中的激活函数改为小波函数(解析形式),或者将二代小波的离散化构成滤波器组集合(包括尺度因子、平移因子和方向因子等,例如脊波、曲波、轮廓波等),随机地选取若干作为网络模型初始化的参数;深度 SVM 网络,即特征学习阶段使用深度神经网络,分类任务阶段使用 SVM 分类器;还有诸如深度极限学习机、深度字典学习、深度 ICA 网络,深度强化学习等,都是从方法层面上进行的组合,形成新的深度神经网络,以期获取简单模块的可控性(例如参数可分层优化避免梯度弥散现象等)和解释性(生物机理)对全局网络模型的影响。本质上,深度学习吸纳了机器学习的框架,稀疏表示学习的特性和优化策略,使得学习到的特征有利于提升网络模型的泛化性能,也丰富了网络训练的各种技巧与策略,更为重要的是将数据的自适应分析进一步推向人工智能。

众所周知,机器学习、稀疏表示学习和深度学习的理念有所不同,机器学习在中小规模的数据上追求精度与效率,所以花费大量的时间研究数据的先验特性并且模型是可拆分的;稀疏表示学习的理念则是在保证输入结构特性不发生严重扭曲的前提下,实现字典和表示系数的学习,模型的设计是线性的且考虑数据的分布特性;而深度学习的理念是在中大规模数据集上追求简单,兼顾精确,不强调数据的先验特性,模型讲求统一的端到端的设计方式。另外,在不同硬件和数据背景的驱动下,应用任务的理念也在发生着重大改变。

16.1.2　深度学习、计算与认知的范式演进

目前,常用的深度学习的网络模型有卷积神经网络(监督学习),前馈神经网络(半监督学习,例如以深度置信网络为代表的堆栈自编码构成的深度堆栈神经网络),生成式对抗网络(无监督学习,包括生成网络与判别网络的设计)。之前,多隐层的感知机(或称深度前馈神经网络)的缺点是:一是有类标数据少,训练不充分,易出现过拟合现象;二是构建的优化目标函数为高度非凸的,参数初始化影响网络模型的性能(因为可行域内出现大量鞍点和局部最小值点),极易陷入局部最优;三是利用反向传播算法,当隐层较多时,由误差反馈其靠近输出端的权值调整较大,但靠近输入端的权值调整较小,出现所谓的梯度弥散现象。针对这三个缺点,提出的改进策略有:一是增加数据量,改进统计方法,如裁剪、取块等;或减少层与层之间的权值连接,间接增加数据量;再或利用生成式对抗网络学习少量数据的内



在分布特性,然后根据采样来扩充数据等;二是逐层学习加精调策略,利用自编码或传统的机器学习方法和稀疏表示/编码方法在无监督学习方式下实现逐层的权值预训练(逐层权值初始化),通过保持层与层之间的拓扑结构特性来避免过早地陷入局部最优;三是为了弱化梯度弥散现象,在初始化的参数上,引入随机梯度下降实现精调来克服输入端的权值未充分训练的问题。

深度学习、计算与认知的范式中,关于"深度"的定义,有时间上(如深度递归神经网络)和空间结构上(如深度卷积神经网络)的区别,其相对应的输入分别为序列向量和图像(或视频),并且计算与认知的形式也有所不同,时间上的深度递归神经网络旨在挖掘序列数据中的上下文逻辑特性,空间结构上的深度神经网络主要挖掘数据中的高层语义特性(层级特征提取);在该范式中,强调非线性的操作(激活函数),即从数据空间到特征空间的扭曲能力;注重的是端到端的设计模式,各种子模块的组合,例如卷积神经网络中的卷积、池化、非线性和批量归一化、全连接和 Softmax 分类器的组合;深度置信网络中的受限玻尔兹曼机、全连接和 Softmax 分类器的组合;深度前馈神经网络中的自编码网络的堆栈(如基于分析合成形式的包括稀疏自编码、降噪自编码、卷积自编码、可收缩性的自编码等,基于合成形式的稀疏编码、卷积稀疏编码等);深度生成神经网络中的生成式对抗网络,其中生成网络和判别网络分别可以用反卷积神经网络和卷积神经网络,或受限玻尔兹曼机和玻尔兹曼机等;深度递归神经网络中的长短时记忆网络的组合(注重隐层回路的设计)。

目前,深度学习的学习方式包括监督、半监督和无监督,其中半监督方式下的逐层学习(大量无类标数据)加精调(少量有类标数据)的模式最为成熟,无监督方式下的深度学习最为新颖,如基于生成式对抗网络的深度生成神经网络(该网络的性能取决于数据量,以及迭代更新判别网络和生成网络的策略),或特征学习加机器学习中的无监督方法(如 K-means 聚类算法)形成的层级聚类特性的深度网络等。深度学习未普及以前,研究人员普遍认为:学习有用的、多级层次结构的、使用较少先验知识进行特征提取的这些方法都不可靠,确切地说是因为简单的梯度下降会让整个优化陷入不好的局部最小解,或者误差在多隐层内反向传播时,往往会发散而不能收敛到稳定状态;但目前,深度学习的核心不再是找到全局最优解,而是近似最优解,随着自编码网络、稀疏编码、生成式对抗网络、小波分析等方法应用于参数的初始化,可以在保持输入的拓扑结构的同时避免过早地陷入局部最优,通常,近似最优解也是实际中的可行解。

16.1.3 深度学习形成脉络

本节我们将以图谱的形式给出各种深度神经网络的关系图,力求从监督、半监督和无监督的角度来阐述网络模型的知识图谱,并给出深度学习的发展趋势和软件基础架构。

下面基于深度学习的知识图谱(图 16.1)和不同学习方式下的典型深度学习网络(图 16.2),我们来给出深度学习的几个关键的历史趋势。首先,"深度学习"有着悠久而且丰富的历史与哲学观点,例如可追溯到 20 世纪 40～60 年代的控制论,20 世纪 80～90 年代的连接机制等;其次,可用的训练数据量迅速增加,以及计算机软硬件基础有所完善,使得

深度学习继 2006 年深度置信网络提出来后又重新变得更加有用,并且各种形式或策略下的深度神经网络如雨后春笋般的增长;再次,深度学习已经能够解决日益复杂的各种实际应用任务,并且精度在不断提高,例如谷歌的 Deepmind 团队开发的 AlphaGo 及其升级版 Master 可用于围棋游戏,它的存在代表着人工智能在面对无法用穷举法策略解决问题时,已经可以通过特定的策略来减少不必要的计算,并且从海量的对弈学习中强化正确的决策;再比如百度的"小度"用于跨年龄人脸识别等。最后,将对人工智能领域产生重大变革的深度学习模型有深度卷积神经网络、深度强化学习、深度递归神经网络和带有博弈策略的生成式对抗网络等,从深(网络层级)到浅(每一隐层的节点个数)的这种架构,力争在刻画数据的内在高层语义特性/逻辑推理特性/分布概型等方面趋近于无监督的学习方式,另外多通路、多尺度策略下的深度学习也将更加趋向人工智能的精髓——模拟人脑多层次性的思考过程。

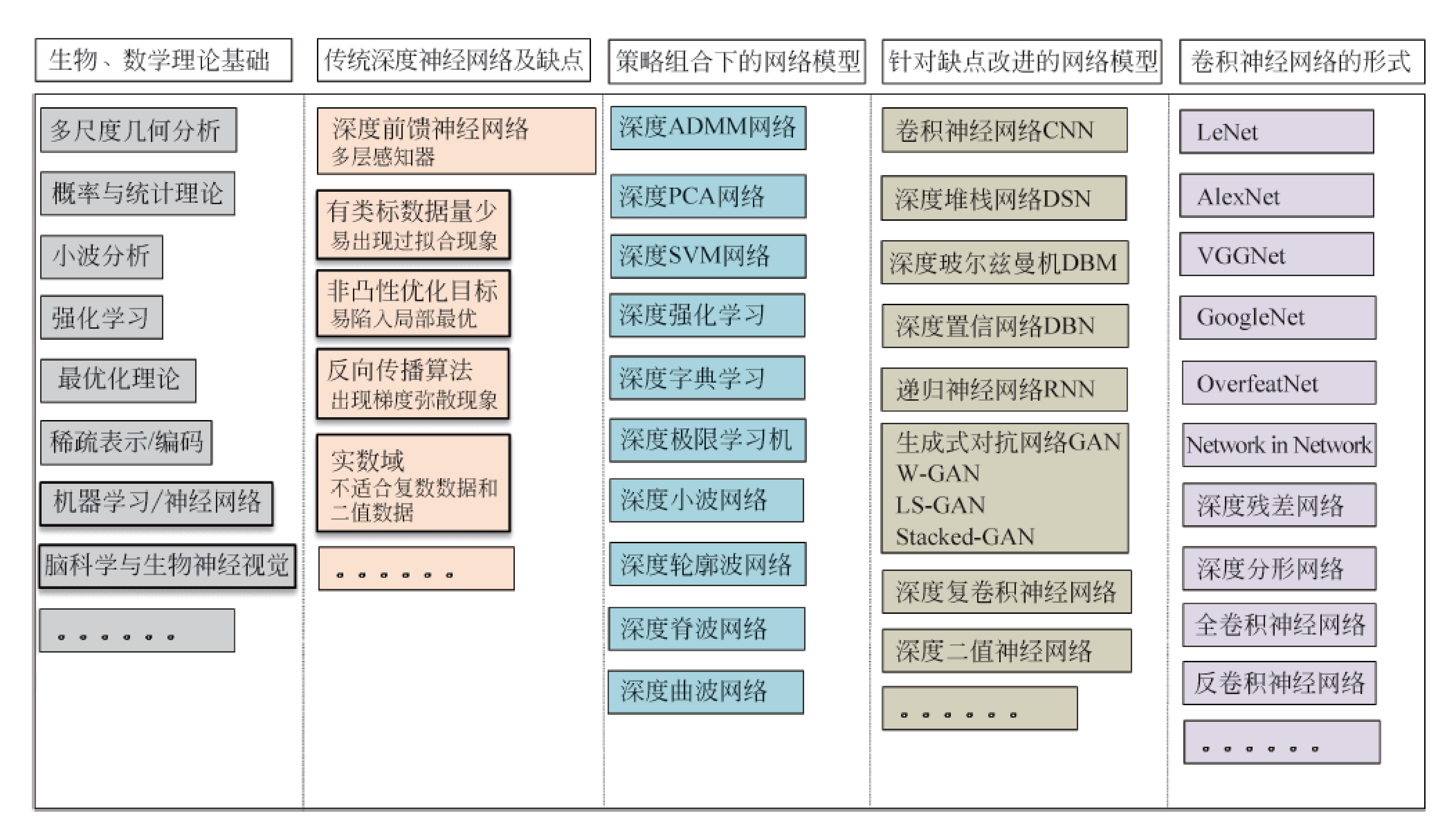
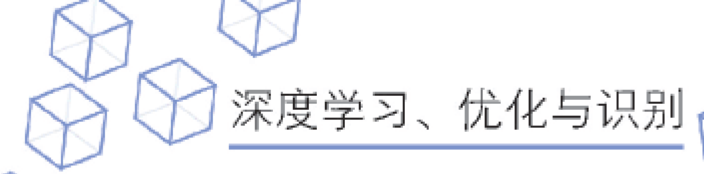


图 16.1　深度学习知识图谱

随着计算机硬件的不断发展,能够有效处理、实时分析大规模数据的各种专用深度学习软件平台也受到人们的高度关注,常用的平台描述总结如图 16.3 所示,常用深度神经网络的结构和模块如图 16.4 所示。

大规模的分布式实现依赖多机的并行化,即由于每个机器上的计算资源往往是有限的,希望把训练的任务分摊到多个机器上进行。不论是数据并行化处理,还是模型并行化交互,抑或是模型的压缩等,从计算机视觉的角度,都直观地给出了深度学习对数据的分析的有效性和合理性,但生物可解释性仍缺乏相应的证据来说明这一点,并且基于深度学习的数学理论分析仍不完善,对于较好的实验结果仍缺乏规律性的描述与总结,目前大多仍是依赖实验经验的指导,譬如超参数的架构、参数调节的技巧等。



有监督学习方式 依据：网络的训练依赖于有类标数据集	半监督学习方式 依据：网络的预训练依赖于大量无类标数据集，精调依赖少量有类标数据集	无监督学习方式 依据：网络的训练仅依赖于无类标数据集
深度卷积神经网络	深度堆栈网络 深度反卷积神经网络	生成式对抗网络 深度卷积生成对抗网络
基本单元——卷积流 即卷积、池化、非线性和批量归一化； 全连接层 分类/回归器设计	基本单元——自编码网络和卷积稀疏编码网络 即稀疏自编码、降噪自编码和卷积自编码等；受限玻尔兹曼机，玻尔兹曼机；稀疏编码	基本模块——生成网络和判别网络 即生成网络和判别网络的设计为多隐层神经网络，且生成网络的输出与真实输入进入判别网络时，尺寸一致
激活函数 修正线性单元	激活函数 修正线性单元，Sigmoid函数和Tanh函数	
求解策略 随机梯度下降	学习策略 逐层学习，用于权值初始化精调，权值(微)更新	学习策略 生成网络、判别网络的权值更新，可以通过交替迭代的方式学习；通常，更新一次判别网络后，需多次更新生成网络
技巧 Dropout，Maxout，稀疏正则		
特性 局部连接、权值共享和平移不变特性		特性 学习自然或真实数据集的分布特性，从分布概型角度可用于扩张数据集
典型应用场景 分类任务、目标检测等	典型应用场景 分类任务、目标检测、图像压缩与降噪等	典型应用场景 聚类、数据生成、超分辨任务等

图 16.2　不同学习方式下的典型深度学习网络

Caffe	Tensorflow	Torch7	Keras—Theano	MxNet
卷积神经网络架构下的快速特征提取	深度神经网络的多维数组流图的计算平台	支持机器学习和深度学习的科学计算框架	基于Theano的深度学习框架，包含各种深度神经网络的模块	轻量化分布式可移植的深度学习计算平台
语言 C++/Cuda架构 支持使用Python和Matlab接口 支持在CPU和GPU上运行与切换	语言 底层语言为C++ 开发语言使用Python 支持在CPU和GPU上运行	语言 底层语言为C++ 脚本语言LuaJTT 支持在CPU和GPU上运行及并行化	语言 C++/Cuda架构 的编写开发使用Cython 支持在CPU和GPU上运行和切换	语言 C++、Python、Julia、Matlab、R、Scala 支持Cuda和云计算以及多GPU、多机节点的计算
典型网络 LeNet、AlexNet Detector	典型网络 生成式对抗网络GAN	典型网络 深度二值神经网络、深度卷积神经网络	典型网络 自编码器 深度堆栈网络DSN 深度置信网络DBN	典型网络 VGGNet、深度复卷积神经网络、递归神经网络RNN

图 16.3　常用深度学习平台的简介

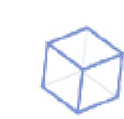


图 16.4　常用深度神经网络的结构与模块



总之，深度学习是一种吸纳了众多学科或其分支的精华、思想的方法，不同学科已存在且基于脑启发式的算法都将对这一方法产生积极/消极的冲击，其中积极的冲击也将显著地完善人脑的认知和思考过程、推动着人工智能的发展，例如深度强化学习等。

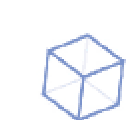
16.2 深度学习的应用介绍

16.2.1 目标检测与识别

目标检测与识别是指从一幅场景(图片)中找出目标，包括检测(主要回答场景有无目标及其所在的位置，即 Where)与识别(识别检测的区域或目标是什么，即 What)两个过程；任务的难点在于待检测区域/候选框的提取与候选框的识别，所以该任务的大框架为：首先，建立场景提取候选框的模型；然后，识别候选框的分类模型；最后，精调分类模型的参数和有效候选框的位置精修。

该任务中，训练数据的形式为：输入场景(图片)，输出为场景中的目标位置与类别；经典的模型为基于区域的卷积神经网络，即 R-CNN；其中候选框提取的阶段：采用选择搜索的策略，对场景输入得到 1k～2k 个不同种类的候选框区域，由于区域大小不一，所以在识别阶段之前，需要缩放处理为一致的尺寸；其中识别的网络采用的是卷积神经网络(如 AlexNet 或 VGG16 提特征，SVM 分类器)；对应着的优化目标函数为两部分，一是识别阶段的损失函数，另一个是候选框位置评估的损失函数，即建立真实框的位置与有效候选框位置(根据二者的交叠面积是否大于某个设定的阈值来判别有效性)的回归器，主要用于有效候选框的位置精修。需要注意的是：R-CNN 模型中提取候选框、识别阶段的特征提取与分类器设计，以及有效候选框的位置精修是分阶段依次进行的，并不是一个统一的框架。

由于 R-CNN 的缺点是在于更多的候选窗并不能提升性能，且存在大量的重叠，以及设计较为松散，存在着大量的操作冗余使得时间和存储复杂度过高。所以为了对 R-CNN 进行改进，先后提出了 fast R-CNN 和 faster R-CNN，其仍沿用 R-CNN 的框架，但不同的是 fast R-CNN 将识别阶段的特征提取、分类器设计以及候选框精修整合为一个深度网络(分类器使用 Softmax)，同时在该深度网络的全连接层使用了提速策略——权值矩阵的奇异值分解，注意在 fast R-CNN 中候选框的提取与识别阶段的深度网络是分阶段进行计算的；而 faster R-CNN 则首次提出使用了候选区域生成网络(注意不再是之前 R-CNN 和 fast R-CNN 中的选择搜索方法)，通过将候选区域生成网络和 fast R-CNN 中识别阶段的深度网络融合形成了一个统一的基于深度网络的目标检测与识别框架(其中这两个网络训练时采用的共享卷积层流的方式)，注意候选区域生成网络的核心思想是通过卷积神经网络直接生成候选区域。此后，全卷积神经网络和显著性检测也都被用于目标检测与识别中的候选区域提取中，形成的仍是一个统一的网络模型。



16.2.2　超分辨

超分辨任务是指将同一场景的低分辨图变为高分辨图的过程，需要模式化该过程，即寻找低分辨图与高分辨图的内蕴特性（例如稀疏表示中低分辨图在低分辨字典的表示系数与高分辨图在高分辨字典下的表示系数是一样的）；任务的难点是未知高分辨图退化为低分辨图的机理，需作出合理的假设（如低分辨率的图完全拥有用于推理预测其所对应的高分辨率部分的信息），来逆向（从低分辨到高分辨）建模倒逼，寻找这种内蕴特性的表达。基于深度学习的超分辨任务的大框架有两种：一种是特征空间中各自所对应的特征映射交互，即低分辨图输入深度神经网络（记为 Low Resolution DNN）得到特征，记为 Low Resolution Feature maps；同理高分辨图输入 High Resolution DNN 网络得到 High Resolution Feature maps，通过建立 Low Resolution Feature maps 与 High Resolution Feature maps 之间的连接，最终形成端到端模型（输入为低分辨图、输出为高分辨图）的统一架构，对应着模型需学习的参数包括 Low Resolution DNN 和 High Resolution Feature maps 两个部分；另一种是在设计低分辨图到高分辨图（预测或生成的）的网络这个过程中，为了利用高分辨图（真实的、训练数据集中的）来指导或监督这个过程的学习（注意不再使用空域的损失函数，即真实与预测的差），需要在预测的高分辨图和真实的高分辨图中作出博弈（可以通过设计二分类深度神经网络来实现，即二类指的是真实的（＋1）、生成的（－1）），期望预测的接近于真实的高分辨图，同时期望预测和真实的高分辨图保持一定的差异性。

该任务中数据的形式为：输入为低分辨图，输出为对应的高分辨图；经典的模型是基于生成式对抗网络的超分辨任务，记为 SuperResolution-GANs；其中生成式对抗网络包括两个部分，生成网络（可用各种深度神经网络，如卷积神经网络、深度置信网络等）与判别网络（带有二分类器的各种深度神经网络）；其中生成网络用于将低分辨图变为高分辨图，也称伪高分辨图；判别网络用于区分真实的高分辨图和伪高分辨图，进而实现指导生成网络的过程；需要注意的是伪高分辨图与真实高分辨图的尺寸是一致的。优化时，需要满足判别网络尽可能以最大的概率判别真伪，同时又需要生成网络得到的伪高分辨图尽可能与真实的高分辨图分布特性一致，即判别网络以最小的概率将生成的伪高分辨图判断为真实的高分辨图。当生成式对抗网络训练完成后，取出生成网络部分便可实现低分辨图到高分辨图的恢复；其中的求解仍采用交替迭代，即固定判别网络时，优化生成网络的参数，同理固定生成网络时，优化判别网络的参数。

16.2.3　自然语言处理

自然语言处理是指创造能够处理或是理解语言以完成特定的任务的系统，这些任务通常包括问答系统（例如 Siri 做的事情），情感分析（判断一句话隐含着的积极或消极的意义）、图片标注（为输入的图像生成一个标题）、机器翻译（将一种语言翻译成另一种语言）、语音识别、词性标注和命名实体识别等。如何建立更为有效的深度神经网络模型来处理自然语言？

该任务的核心技术是词向量(即通过该向量值来表征词的语境和语义,记为 Word2Vec)和循环/递归神经网络、门递归单元、长短时记忆网络等,不同任务下的网络模型不同,例如基于递归神经网络的问答系统,情感分析的树状长短时记忆网络,基于深度 LSTM 的机器翻译等。

下面我们简要地介绍深度递归神经网络用于问答系统的框架,如果我来问你一个问题"RNN 表示什么",如果学习过,你就一定能告诉我答案。这是因为你通过阅读、存储记忆,已经吸收了这些知识,你只需要简单地花几秒钟去定位这条信息,然后把它用通顺的语言表达出来。如何量化该过程并设计网络?首先,将输入转变成一个特征,这需要用到词向量、词性标注、解析等;其次,依据当前的特征,对过去的记忆进行更新,从而反馈系统已经接收到的新输入;再次,根据更新的记忆对输入的特征进行表征,得到新的特征;最后,对新特征进行解码输出反馈,这一个过程可理解为将记忆的新特征转化为一个可读的、准确的问题答案,实现该过程的核心网络便是深度递归神经网络。注意训练的方式为监督训练,训练数据包括原始文本、问题、支撑句以及基底真实答案等。

众所周知,深度递归神经网络是一类专门处理序列数据的模型,而自然语言处理的核心就是序列标注与逻辑推理;目前,基于各种改进的长短时记忆网络的自然语言处理任务层出不穷,应用驱动下的目标包括:咨询、售后服务聊天机器人、完美的实时机器翻译系统,以及掌握对无结构文本或长文本更深的理解能力的问答系统。

16.3 深度神经网络的可塑性

16.3.1 旋转不变性

众所周知,大脑对物体的认知或识别具有一定角度下的旋转不变性;从仿生学的角度,模拟大脑计算的深度学习是否具有旋转不变性?实验(数据分为训练数据集和测试数据集,网络模型为深度神经网络,利用训练数据集充分训练深度神经网络,得到模型参数;进而对测试数据集(旋转不同的角度)进行测试并统计准确率)证实,深度神经网络在(顺时针或逆时针)旋转较小角度的测试数据集上具有相对较高的识别准确率,即具有一定的旋转不变性。例如,对于深度卷积神经网络,由于旋转特性主要体现在卷积的操作上(当然部分地也体现在池化和非线性的操作上),即卷积核的设计应包括局部化、方向和多尺度特性,其中的方向特性类似于生物视觉腹侧视觉通路中的感受野接收范围,随着网络模型中的层次加深,这种方向特性也应逐渐增大。但是,在实验中,为了对较大旋转角度的测试数据集保持较高的识别准确率,通常需要将各种角度下的数据集进行混合来扩充训练样本集,实现网络的重新训练;这样处理带来的好处是:扩充后的数据集,使得网络模型能够得到充分的训练,同时可以预防欠拟合现象的发生。

谷歌的 Deepmind 提出了空间变换神经网络,针对卷积神经网络的特点,构建了一个新的局部网络层,称为空间变换层,它能将输入图像做任意空间变换(即将平移、裁剪、伸缩、旋





转这四种操作用于一种统一的结构),进而提升深度神经网络的旋转不变特性。

16.3.2　平移不变性

什么是平移不变性?对于深度神经网络模型而言,是指输入图像整体上发生了一定的平移,但照样能够提取特征进行正确的识别,即不影响输出的响应。目前,实验表明,深度卷积神经网络中的池化操作具有一定的平移不变性,能够更好地描述物体的各个部分的几何形变,即如果选择图像中的连续范围作为池化区域,并且只是池化相同或重复的隐藏单元产生的特征,那么,这些池化单元就具有平移不变性,换言之,图像经历了一个小的平移之后,依然会产生相同的(池化后的)特征;这对于很多的应用任务,如分类、物体检测、声音识别等,都希望得到具有平移不变性的特征,因为即使图像经过了平移,输入(图像)的类标仍然保持不变。例如,如果你处理一个 MNIST 数据集的数字,把它向左侧或右侧平移,那么不论最终的位置在哪里,都期望分类器仍然能够精确地将其分类为相同的数字。

池化对于空间区域具有平移不变性(备注:局部平移不变性是一个重要的性质,尤其当我们关心某个特征是否出现而不关心它出现的具体位置时);应用实践对于在不同的情况下应当使用哪种池化函数给出了一些指导,例如无论是最大池化还是平均池化都是在提取区域特征,均相当于一种抽象,抽象就是过滤掉了不必要的信息(当然也会损失信息细节),所以在抽象层次上可以进行更好的识别。二者效果的差异通常不会超过 2%,由于评估特征提取的误差主要来自两个方面:一是邻域大小造成的估计值方差增大,平均池化能减小这种误差;二是卷积层参数误差造成估计均值的偏移,最大池化能减小这种误差。总之,平均池化对背景保留更好,最大池化对纹理提取更好。

16.3.3　多尺度、多分辨和多通路特性

目前,更多的深度神经网络将多尺度、多分辨和多通路等特性融入至网络的构造过程中,期望通过软硬件的并行化来提升网络的性能和训练的高效性。其中的核心便是多分辨特性,随着深度神经网络的层级加深,良好的拓扑结构对应性取决于输入样本的分辨率,不同分辨率下的样本差异性对网络模型的性能都有显著的影响,换言之,数据的分级处理体现着输入与输出之间映射的差异性,犹如大脑的多分辨特性,对信息结构完整或分辨率高的输入识别精度高,相反,对结构缺失或分辨率较低的输入识别精度低;若将这种多分辨特性与深度卷积神经网络相结合,形成多分辨深度神经网络,便可以从广度(即多通路性)上削弱对深度(隐层个数,例如极深网络等)的约束与要求,从而改善差异性数据集学习到的深度神经网络的性能,通常的网络结构如图 16.5 所示。

同时,这种多尺度、多分辨和多通路特性可以为深度神经网络带来良好的旋转不变性、伸缩不变性和平移不变性等特性,如图 16.6 所示。

众所周知,小波具有良好的多分辨特性、通过构造嵌套空间序列来实现对输入信号的刻画,以期获得在不同分辨率下的编码有较好的拓扑结构对应和统计特性,如平移、伸缩和旋

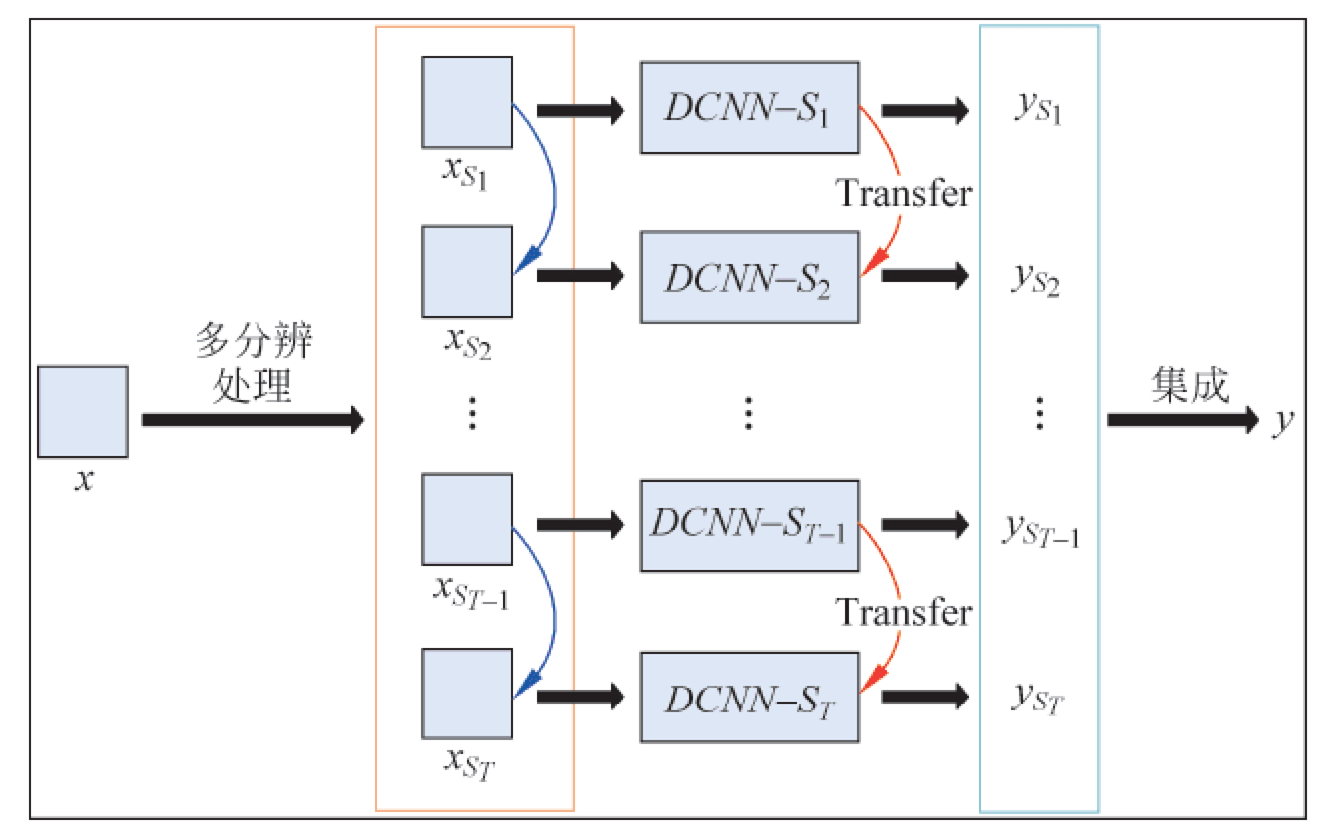


图 16.5　多分辨深度卷积神经网络的结构

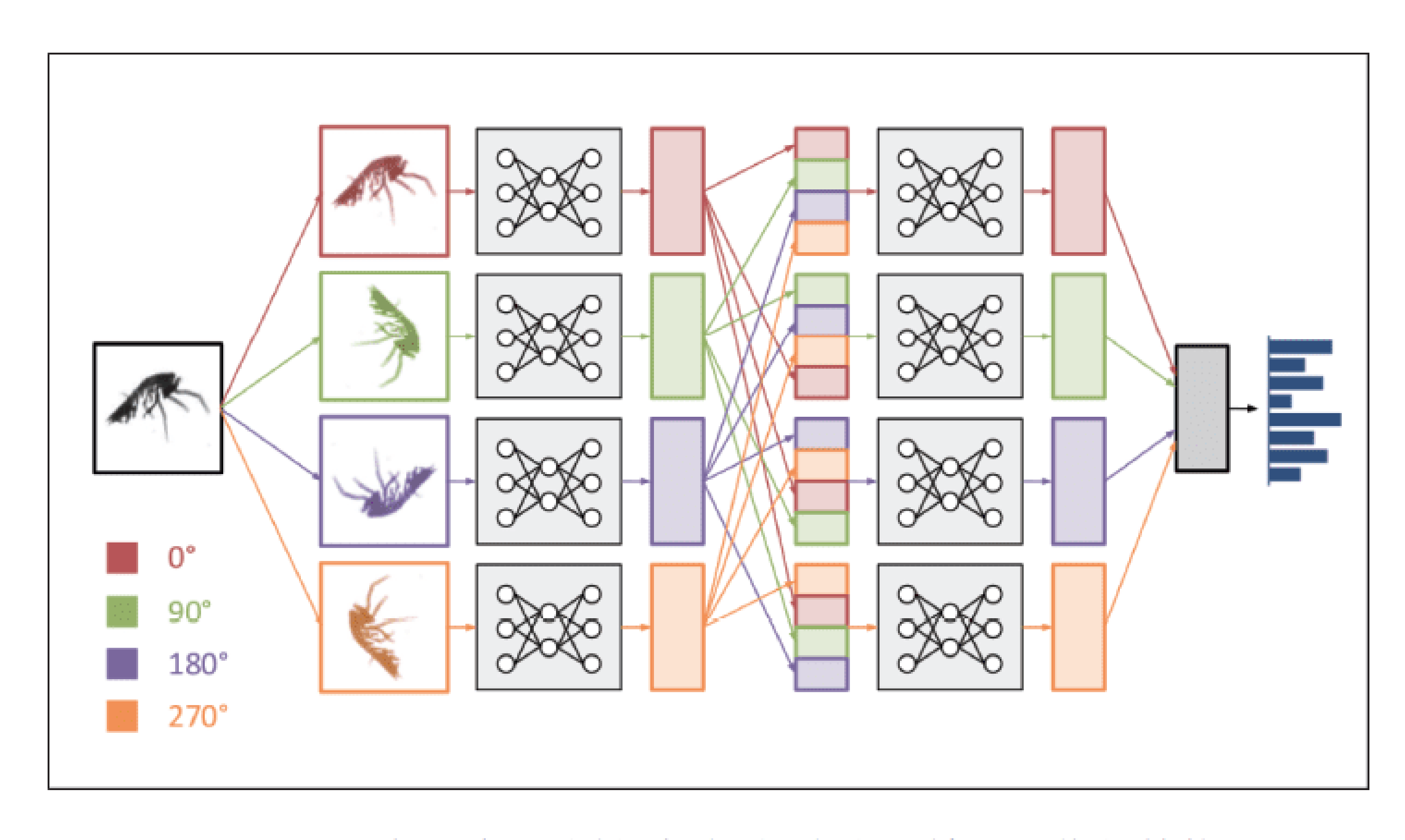


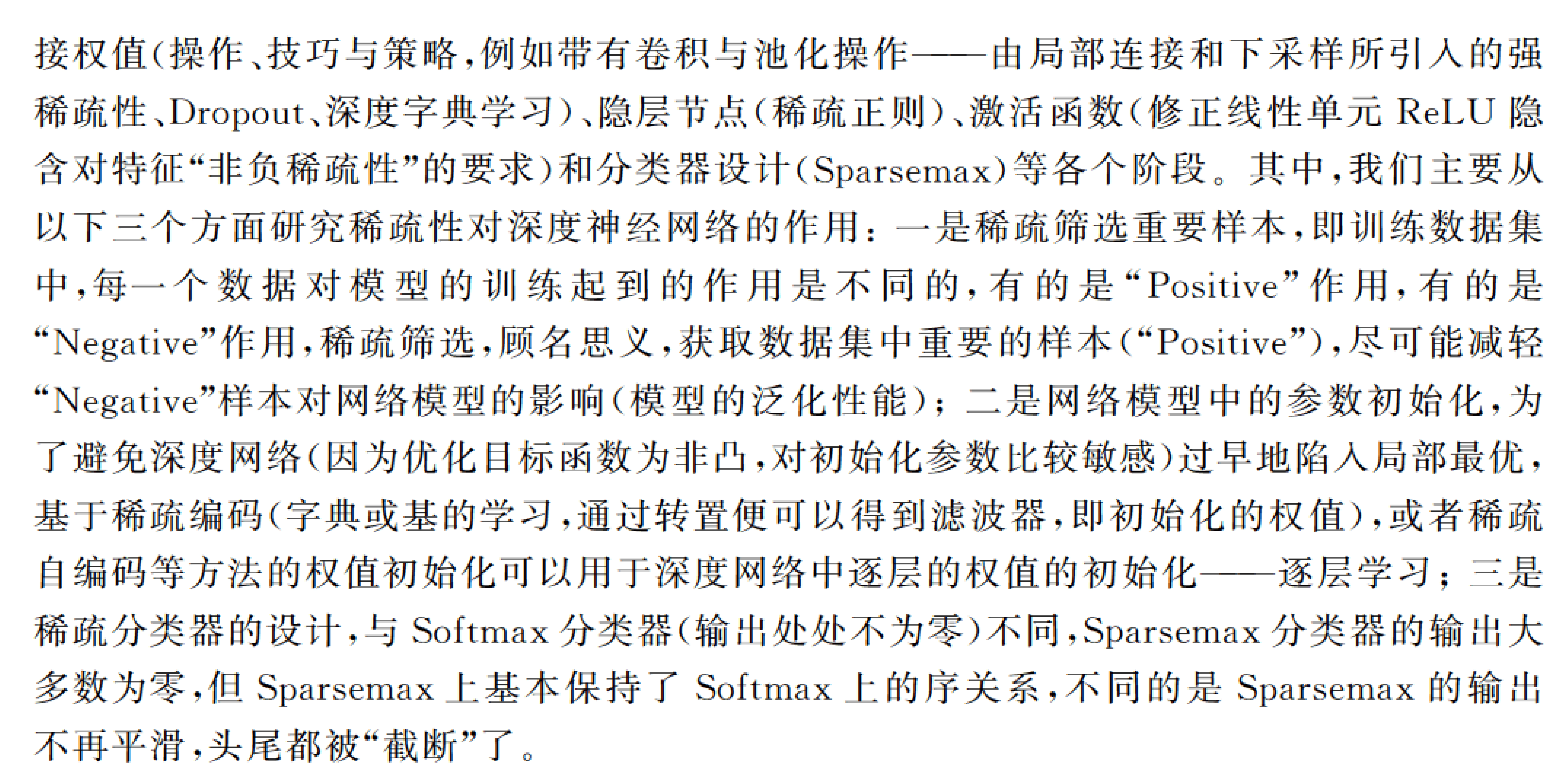
图 16.6　多通路(不同的角度)深度卷积神经网络的结构

转不变特性，其中由嵌套空间所带来的多分辨特性可以聚焦至输入信号的任意细节。目前，基于小波分析或多尺度几何分析的深度神经网络的建模也受到了极大的关注，形成深度小波网络、深度脊波网络、深度轮廓波网络等；本质上，期望所构建的模型具有更好地对输入获取分布式特征的表达能力，以及基于软硬件并行化提升学习的效率。

16.3.4　稀疏性

稀疏性包含两方面内容，一方面指的是信息表示的普遍属性，即信息的表示由大量编码系数中的少量编码系数决定的现象，另一方面指的是模型中所蕴含的稀疏特性，通常认为，一个稀疏网络模型具有很好的解释性——符合生物视觉认知机理，并且网络具有很好的泛化性能。对于深度卷积神经网络模型而言，稀疏性可以贯穿于数据(稀疏筛选重要样本)、连





接权值(操作、技巧与策略,例如带有卷积与池化操作——由局部连接和下采样所引入的强稀疏性、Dropout、深度字典学习)、隐层节点(稀疏正则)、激活函数(修正线性单元 ReLU 隐含对特征“非负稀疏性”的要求)和分类器设计(Sparsemax)等各个阶段。其中,我们主要从以下三个方面研究稀疏性对深度神经网络的作用:一是稀疏筛选重要样本,即训练数据集中,每一个数据对模型的训练起到的作用是不同的,有的是“Positive”作用,有的是“Negative”作用,稀疏筛选,顾名思义,获取数据集中重要的样本(“Positive”),尽可能减轻“Negative”样本对网络模型的影响(模型的泛化性能);二是网络模型中的参数初始化,为了避免深度网络(因为优化目标函数为非凸,对初始化参数比较敏感)过早地陷入局部最优,基于稀疏编码(字典或基的学习,通过转置便可以得到滤波器,即初始化的权值),或者稀疏自编码等方法的权值初始化可以用于深度网络中逐层的权值的初始化——逐层学习;三是稀疏分类器的设计,与 Softmax 分类器(输出处处不为零)不同,Sparsemax 分类器的输出大多数为零,但 Sparsemax 上基本保持了 Softmax 上的序关系,不同的是 Sparsemax 的输出不再平滑,头尾都被“截断”了。

深度学习与稀疏性的关系深刻且本质,不仅有效地模拟了生物视觉的认知机理,而且从算法的层面上大量的尝试是合理且良好的。但需要注意的是过度地引入稀疏正则将会导致网络模型的泛化性能降低。

16.4　基于脑启发式的深度学习前沿方向

16.4.1　生物神经领域关于认知、识别、注意等的最新研究进展

生物视觉皮层如何实现对外界刺激的稀疏响应与目标识别一直是视觉神经科学领域中的一个关键问题。在过去的几十年里,已有些科学家们利用视觉神经生理研究中所获得的实验数据,建立了一些计算模型,在对这一关键问题的探索研究中,取得了较好的成果。如 1969 年 Willshaw 和 Buneman 等人提出的基于 Hebbian 局部学习规则的稀疏表示模型,其中的稀疏表示可以使得记忆能力最大化,进而有利于网络结构中联想机制的建立;1972 年 Barlow 等人给出“稀疏性和自然环境的统计特性之间存在着某种相关性联系”的推论,利用该推论,1996 年 Olshausen 和 Field 提出了稀疏编码,验证了自然图像经过稀疏编码后,学习得到的基函数可以近似描述 V1 区上简单细胞的感受野的响应特性。进一步,随着生物视觉皮层中关于稀疏性研究的不断深入,如从 V1 区上简单细胞的感受野特性逐渐发展到 V1 区上复杂细胞的感受野特性,再到近年来依据自然场景的不同特征(如空间位置、形状、颜色、运动等)将视觉皮层分成不同的视觉通路(见图 16.7,如腹侧视觉通路与背侧视觉通路)实现并行处理的研究,都已经取得了较好的研究成果;对应着生物视觉稀疏认知机理方面所取得的研究进展,稀疏认知计算模型也从稀疏编码发展到了结构化稀疏模型和判别性稀疏模型等,再到近两年来关注的层次化稀疏模型。针对生物视觉皮层如何识别复杂场景下的目标,科学家们结合生物视皮层生理研究的成果,建立了视觉计算模型来对这种识别机

理进行模拟并给出合理地解释。最具代表性的工作有 1999 年美国麻省理工学院人工智能实验室的 Riesenhuber 和 Poggio 提出了层次目标识别模型(记为 HMAX),经验证该模型能够较为合理地解释哺乳动物腹侧视觉通路的信息处理机理;随后 2007 年 Serre 等人扩展了 HMAX 模型,通过引入特征编码字典,使得改进后的模型可以定量地模拟灵长类动物视觉皮层腹侧通路的信息处理机理;进一步,2014 年清华大学胡晓林博士等人在 HMAX 模型的基础上,通过引入稀疏正则的约束,使得复杂场景下的目标识别任务在保证高性能的前提下,其处理的速度,较之前的 HMAX 模型提升了许多。

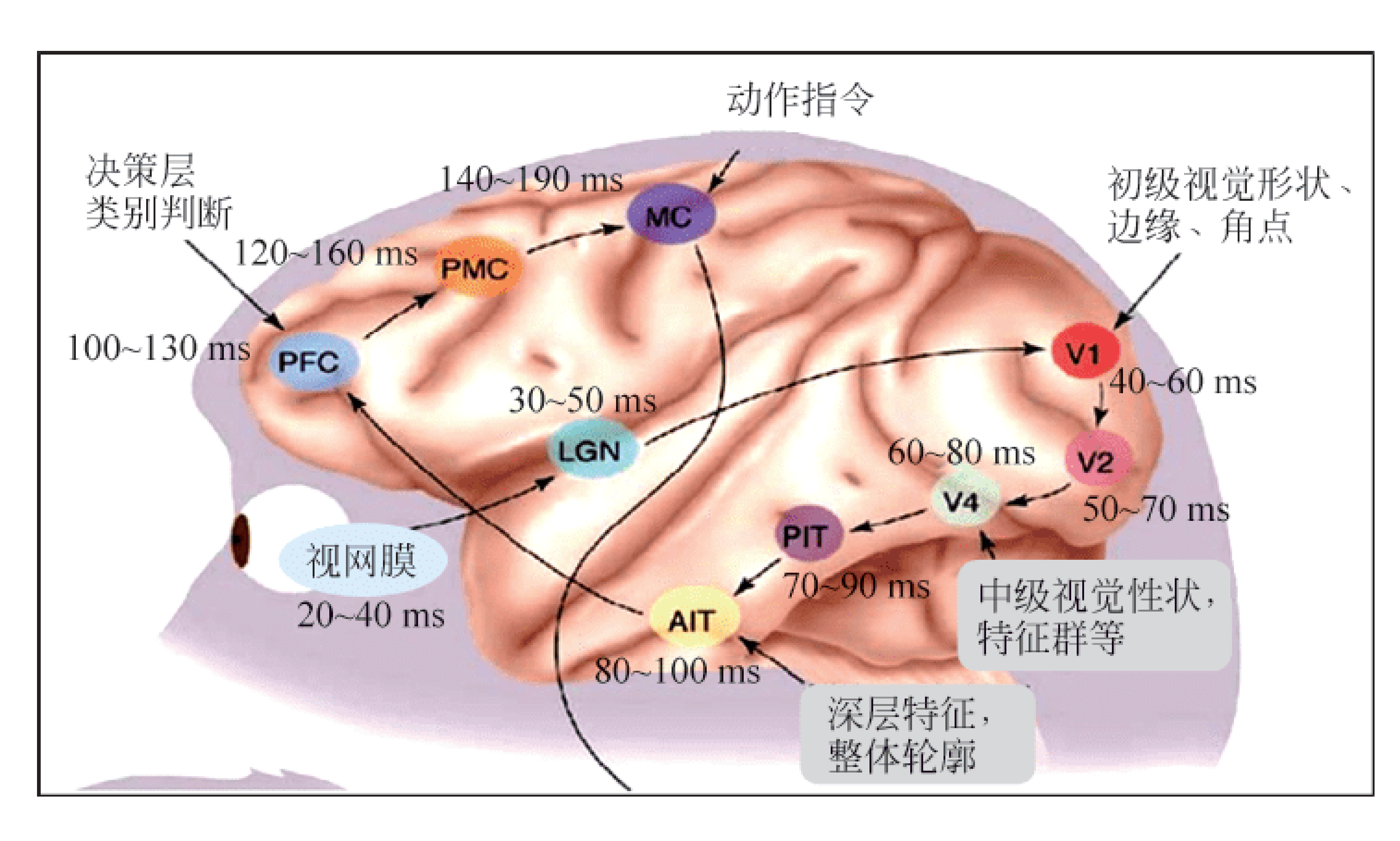
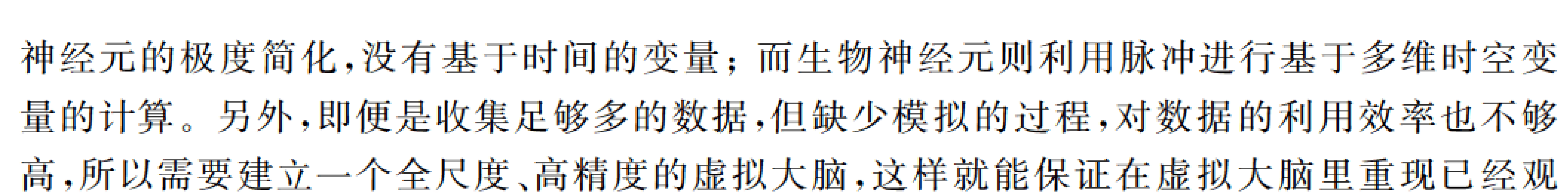


图 16.7 生物视觉系统的信息分级处理

另外,生物神经科学已经揭示了认知行为神经基础的一些基本原理:不同认知行为是由脑内不同的神经环路负责,需要各脑区内的局部神经环路与脑区间长程神经环路的协同工作;学习与记忆是许多认知功能的必要基础,这是由神经细胞之间突触联结的强度与结构的可塑性介导;神经调质可以在多个尺度上调节神经网络的活动与可塑性,从而调控认知行为。在介观层面,科学家们需要描绘脑区之间细胞类型特异性的联结图谱;绘制认知功能的大脑功能图谱;利用因果性手段、揭示认知功能的核心脑区;操控不同脑区及脑区间联结的活动,进而观察认知行为的改变和其他参与环路的活动变化,从而获得脑整体动态规律。在微观层面,需要阐明不同脑区有哪些特定类型的神经元;揭示不同类型神经元是如何参与特定认知功能的;解析不同类型的神经元是如何联结以及这些联结是如何在认知行为中发生动态改变的。这些介观与微观研究将为理解宏观认知行为的神经基础提供重要线索,对于破解人类智能这一终极奥秘具有重要意义。同时,揭示认知行为的神经机制有助于推动脑启发的智能技术(深度学习)的发展。

基于生物神经研究的下一代深度学习,以降低深度学习对数据量的需求为目的,实现小样本学习甚至 One-Shot Learning。需注意深度神经网络确实从生物神经科学领域的研究中获取了一些灵感,但其工作原理与大脑是截然不同的,例如神经元的类型,深度神经网络用的是点神经元,即把信号加权平均的结果输入到一个非线性函数,这种点神经元是对生物

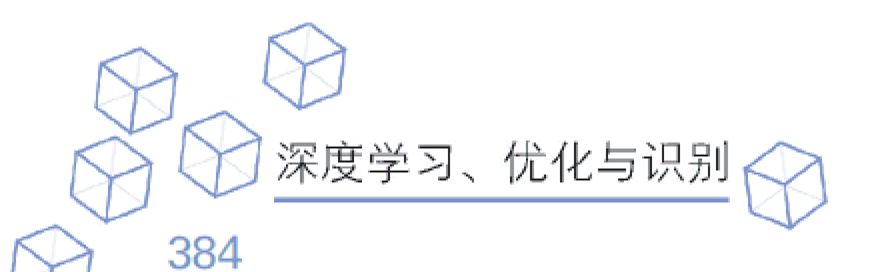



神经元的极度简化，没有基于时间的变量；而生物神经元则利用脉冲进行基于多维时空变量的计算。另外，即便是收集足够多的数据，但缺少模拟的过程，对数据的利用效率也不够高，所以需要建立一个全尺度、高精度的虚拟大脑，这样就能保证在虚拟大脑里重现已经观察到的大脑的现象和特征，从而更加准确地提出测试各类神经元的计算模型。

16.4.2　深度神经网络的进一步研究方向

目前，针对深度学习的研究主要集中在以下三个部分：一是理论算法（关于算法的稳定性与收敛性；其中稳定性是针对对抗样本的抗干扰能力，收敛性是参数随着迭代次数的变化，逐渐趋于平衡的状态）的研究，针对过拟合现象、梯度弥散现象等导致的模型泛化性能差，本质上，是由于深度神经网络的优化算法鲁棒性差，即输入与输出的关系（非线性利用渐进线性描述）是病态的，其中的正则化约束并未使得关系良态化，另外非凸性的优化目标函数（即可行域中存在着大量的鞍点与局部极值点）对于参数的初始化十分敏感，基于逐层优化的策略（例如自编码、受限玻尔兹曼机、生成式对抗网络、稀疏编码、独立成分分析等）可以较好地解决参数初始化的问题（好的初值本质上加速收敛进程），但预训练后的网络仍存在着收敛性难的问题，换言之，即使精调后的网络的收敛性与稳定性仍依赖有类标数据的量，依据大数定律，当有类标的数据量较多时，网络模型的训练较为充分，理论上稳定性与收敛性置信度更高；例如 Ian Goodfellow 等人提出的生成式对抗网络框架，并给出了严格的稳定性与收敛性的数学证明；对于深度神经网络何时收敛，如何取得较好地局部极小点，每一层变换取得了哪些对识别有益的不变性，又损失了哪些信息等，Mallat 利用小波对深层网络结构进行了量化分析，是在这一个方向上的重要理论探索；二是应用推广（包括基于递归神经网络的自然语言处理、基于生成式对抗网络的超分辨任务、基于 faster R-CNN 目标检测与识别和基于卷积神经网络的人脸识别等）催生的一些优秀的深度学习模型与训练技巧，当前较为成功的应用有谷歌的 Deepmind 团队研发的 AlphaGo 及其升级版 Master 横扫围棋界顶尖职业棋手，背后的核心技术为深度强化学习（强化学习与深度卷积神经网络的结合）；百度深度学习研究院研发的机器人“小度”也在跨年龄人脸识别任务中险胜世界记忆大师和亚马逊开发的用于无人超市值守的 Amazon Go，进一步将人工智能及其背后的核心技术——深度卷积神经网络推向高潮；三是基于生物神经启发的深度学习改进，如基于生物神经元而利用脉冲进行基于多维时空变量的计算而提出的深度脉冲神经网络，可以有效降低算法的复杂度（时间和存储复杂度）。

当前，深度学习技术主要是数据驱动的，即对一个特定任务来说，只要增加训练数据的规模，深度学习模型的表现就可以得到提高。但是发展到今天，这种思路面临很多挑战。主要面临下面几个问题：一是很多领域，很难获取大量的监督数据或者数据的标注成本过高；二是训练数据规模再大，也有难以覆盖的情况。例如聊天机器人，我们不可能穷尽所有可能的答案，而且很多答案也是随时间变化的；因此仅仅依靠大规模的训练语料，并不能解决这些问题；三是通用深度学习模型，直接应用到具体问题，表现（效果、性能、占用资源等）可能不尽如人意；这就要求根据特定的问题和数据，来定制和优化深度学习网络结构，这个是当



前研究最多最热的地方；四是训练的问题。包括网络层数增加带来的梯度衰减，如何更有效地进行大规模并行训练等。为了解决这些问题，当前的研究前沿主要包括以下几个方向：一是引入外部知识，例如知识图谱等；二是深度学习与传统方法的结合，包括人工规则与深度神经网络的结合、贝叶斯与深度神经网络的结合、迁移学习与深度神经网络的结合、强化学习与深度神经网络的结合和图模型与深度神经网络的结合等；三是无监督的深度生成模型；四是新的网络结构新的训练方法等。

16.4.3 深度学习的可拓展性

深度学习不仅注重生物神经的可解释性，而且网络模型的物理可描述性（层级架构的组合，超参数的选取等）和数学优化与求解同样重要；无论是对偶学习还是预测学习，或是无监督学习等，本质上，这些学习范式都将削弱网络模型对有类标数据的依赖性，带有“感”、“知”、“用”、预测、记忆与遗忘、推理特性的深度神经网络将进一步解释、模拟大脑学习、认知与计算的过程。当前，亟须研究的是深度学习的数学基础，核心问题：对于深度神经网络，量化输入与输出之间函数（非线性算子）的奇异性，根据算子的谱范数，判断该系统的稳定性与收敛性；其次，深度学习的表达能力体现在线性与非线性操作复合下的层级组合策略，是否层级越深，表达能力越强？另外，需回答两个问题，一是不增加数据量的情形下，利用格式搜索超参数所构建的深度网络模型，在误差限一致时，是否存在最优的模型（根据模型的泛化能力和分布式特征的表达能力）来合理地反馈输入与输出之间的关系？二是固定网络模型的情形下，随着数据量的增加，网络性能（包括训练性能和泛化性能）对数据能否趋于饱和状态，除了数据的量，数据的质对网络模型的影响如何？

值得关注的是：数据驱动下的深度学习技术，不仅注重数据的量，而且也注重数据的质，数据的差异性对深度学习的影响是至关重要的，譬如分类任务、类内的聚集特性（越强，说明相似度越高，即共性特征为主，个性化特性为辅）和类间的疏散特性（越大，说明类与类之间的差异性越明显，个性化特性为主，共性特征为辅），对特征设计阶段的权值参数有判别特性，即类内强调共性，类间注重个性；满意度最高的模型状态也间接说明二者（共性与个性）是矛盾统一的。

参考文献

[16.1] Hubel D H, Wiesel T N. Receptive fields of single neurones in the cat's striate cortex[J]. Journal of Physiology, 1959, 148(3): 574.

[16.2] Mallat S G, Zhang Z. Matching pursuits with time-frequency dictionaries[J]. IEEE Transactions on Signal Processing, 1993, 41(12): 3397-3415.

[16.3] Willshaw D J, Buneman O P, Longuethiggins H C. Non-holographic associative memory[J]. Nature, 1969, 222(5197): 960.

[16.4] Barlow H B. Single units and sensation: a neuron doctrine for perceptual psychology? [J].



Perception, 1972, 38(4): 795-798.
[16.5] Olshausen B A, Field D J. Emergence of simple-cell receptive field properties by learning a sparse code for natural images[J]. Nature, 1996, 381(6583): 607.
[16.6] Jenatton R, Obozinski G, Bach F. Structured Sparse Principal Component Analysis[J]. Journal of Machine Learning Research, 2009, 9(2): 131-160.
[16.7] Jenatton R, Audibert J Y, Bach F. Structured Variable Selection with Sparsity-Inducing Norms [J]. Journal of Machine Learning Research, 2010, 12(10): 2777-2824.
[16.8] Duarte M F, Eldar Y C. Structured Compressed Sensing: From Theory to Applications[J]. IEEE Transactions on Signal Processing, 2011, 59(9): 4053-4085.
[16.9] Zhang Q, Li B. Discriminative K-SVD for dictionary learning in face recognition[C]. Computer Vision and Pattern Recognition. IEEE, 2010: 2691-2698。
[16.10] Serre T, Wolf L, Bileschi S, et al. Robust Object Recognition with Cortex-Like Mechanisms[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2007, 29(3): 411-426.
[16.11] Hu X, Zhang J, Li J, et al. Sparsity-regularized HMAX for visual recognition. [J]. Plos One, 2014, 9(1): e81813.
[16.12] Riesenhuber M, Poggio T. Hierarchical models of object recognition in cortex[J]. Nature Neuroscience, 1999, 2(11): 1019-25.
[16.13] Zhao Song Nian. Yao Li, Jin Zhen, Xiong Xiao Yun, Wu Xia, Zou Qi, Yao Guo Zheng, Cai Xiao Hong, Liu Yi Jun, The Whole Video features of sparse representation in the human primary visual cortex: Evidence of brain functional imaging. Chinese Science Bulletin, 2008, 53(11): 1296-1304(in Chinese).
[16.14] Houweling A R, Brecht M. Behavioural report of single neuron stimulation in somatosensory cortex[J]. Nature, 2008, 451(7174): 65-68.
[16.15] Zhou Yang, Wang Jian. The functional organization of the visual cortex and progress from fMRI. Progress in Modern Biomedicine, 2006, 6(9): 79-81(in Chinese).
[16.16] Yao Xing Zhong, Lu Tong Wei, Hu Han Ping. Object recognition models based on primate visual cortices: a review. Pattern Recognition and Artificial Intelligence, 2009, 3(4): 581-588(in Chinese).
[16.17] Li Yi Ping, Zhao Dan Dan, Ma Lin, Liu Yi. Research advances on receptive field. WEST CHINA MEDICAL JOURNAL, 2008, 23(3): 640-641(in Chinese).
[16.18] Shang Li. A study of sparse coding algorithm and their application[D]. Hefei: University of Science of Technology of China, 2006(in Chinese).
[16.19] Jiao Li Cheng. Neural network computation. Xi'an: Xidian university press, 1993(in Chinese).
[16.20] Jiao Li Cheng. Application and Realization of neural network. Xi'an: Xidian university press, 1993(in Chinese).
[16.21] Jiao Li Cheng. Neural Network System Theory. Xi'an: Xidian university press, 1990(in Chinese).
[16.22] Mitchison G. The organization of sequential memory: sparse representations and the targeting problem. Organization of Neural Networks: VCH Verlags-Gesellschaft. Weinheim, Germany, 1988: 347-367.
[16.23] Rolls E T, Treves A. The relative advantages of sparse versus distributed encoding for associative neuronal networks in the brain[J]. Network Computation in Neural Systems, 1990, 1

(4)：407-421.

[16.24] Hyvärinen A，Hoyer P O. A two-layer sparse coding model learns simple and complex cell receptive fields and topography from natural images.[J]. Vision Research，2001，41(18)：2413.

[16.25] Olshausen B A，Field D J. Sparse coding with an overcomplete basis set：a strategy employed by V1？[J]. Vision Research，1997，37(23)：3311-3325.

[16.26] Theriault C，Thome N，Cord M. HMAX-S：Deep scale representation for biologically inspired image categorization[C]. IEEE International Conference on Image Processing. IEEE，2011：1261-1264.

[16.27] Baddeley R，Abbott L F，Booth M C，et al. Responses of neurons in primary and inferior temporal visual cortices to natural scenes.[J]. Proceedings of the Royal Society B Biological Sciences，1997，264(1389)：1775.

[16.28] Carlson E T，Rasquinha R J，Zhang K，et al. A sparse object coding scheme in area V4[J]. Current Biology Cb，2011，21(4)：288-293.

[16.29] Quiroga R Q，Reddy L，Kreiman G，et al. Invariant visual representation by single neurons in the human brain[J]. Nature，2005，435(7045)：1102-7.

[16.30] Candes E J，Romberg J，Tao T. Robust uncertainty principles：exact signal reconstruction from highly incomplete frequency information. Nformaon Hory Ranaon on，2006，52(2)：489-509.

[16.31] Chartrand R. Exact Reconstruction of Sparse Signals via Nonconvex Minimization[J]. IEEE Signal Processing Letters，2007，14(10)：707-710.

[16.32] Chartrand R，Staneva V. Restricted isometry properties and nonconvex compressive sensing[J]. Inverse Problems，2008，24(3)：657-682.

[16.33] Candès E J，Wakin M，Boyd S. Enhancing sparsity by reweighted L1 minimization. Journal of Fourier Analysis and Applications，2008，14(1)：877-905.

[16.34] Elad M. Sparse and Redundant Representations：From Theory to Applications in Signal and Image Processing[M]. Springer Publishing Company，Incorporated，2010.

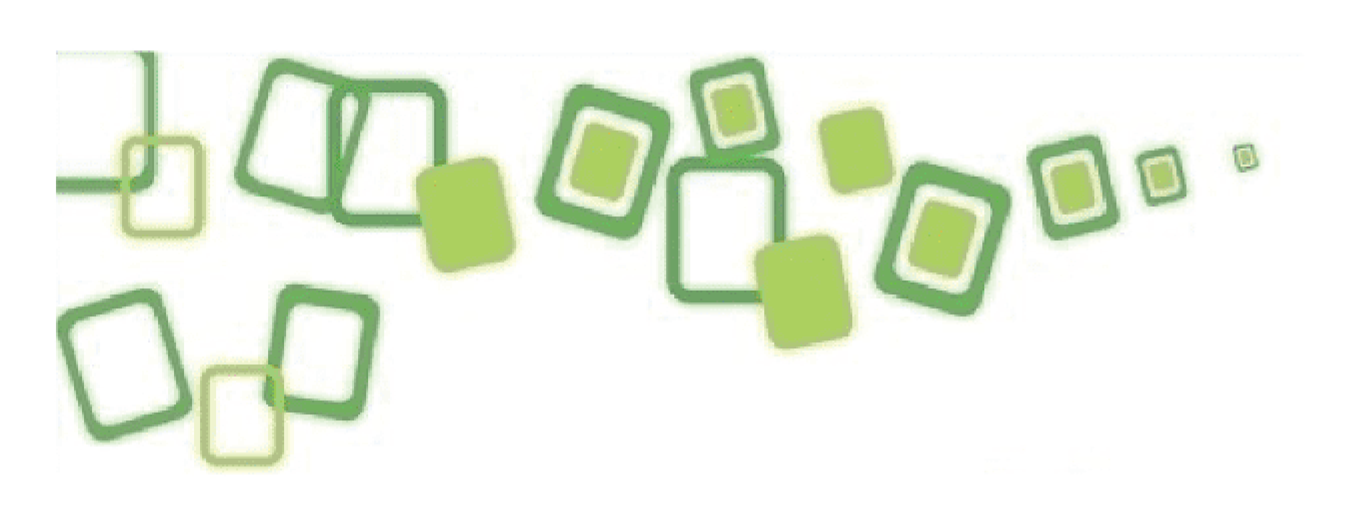
[16.35] Xu H，Mannor S，Caramanis C. Sparse algorithms are not stable：A no-free-lunch theorem[C]. Communication，Control，and Computing，2008，Allerton Conference on. IEEE，2008：1299-1303.

[16.36] Xu Zong-Ben，Zhang Hai，Wang Yao，Chang Xiang-Yu，Liang Yong. L_(1/2) regularization [J]. Science China(Information Sciences)，2010，53(6)：1159-1169.

[16.37] Mi T，Li S，Liu Y. The ℓ 1，analysis approach by sparse dual frames for sparse signal recovery represented by frames[C]. IEEE International Symposium on Information Theory Proceedings. IEEE，2012：2037-2041.

[16.38] Nam S，Davies M E，Elad M，et al. The cosparse analysis model and algorithms ☆[J]. Applied & Computational Harmonic Analysis，2011，34(1)：30-56.

附录A

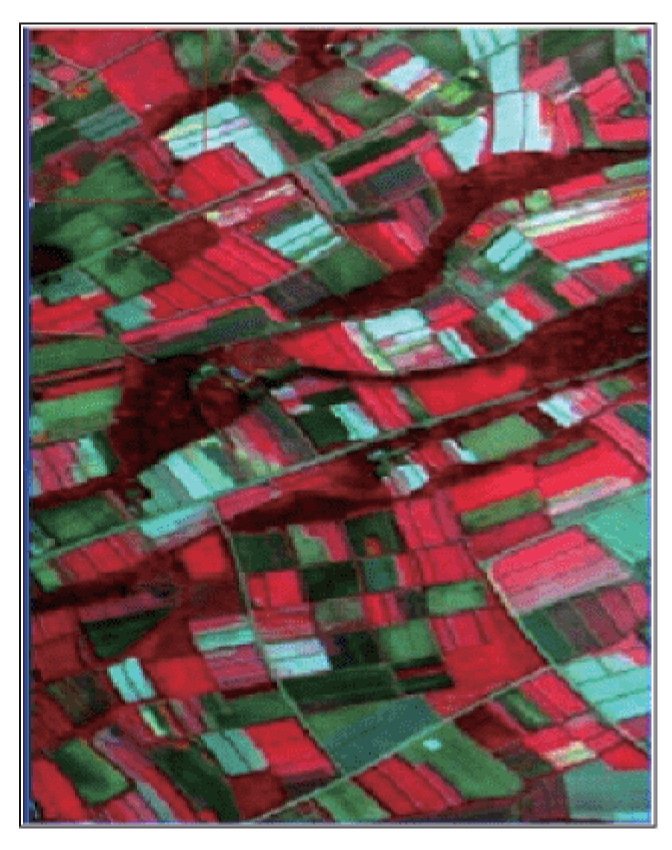


基于深度学习的常见任务处理介绍

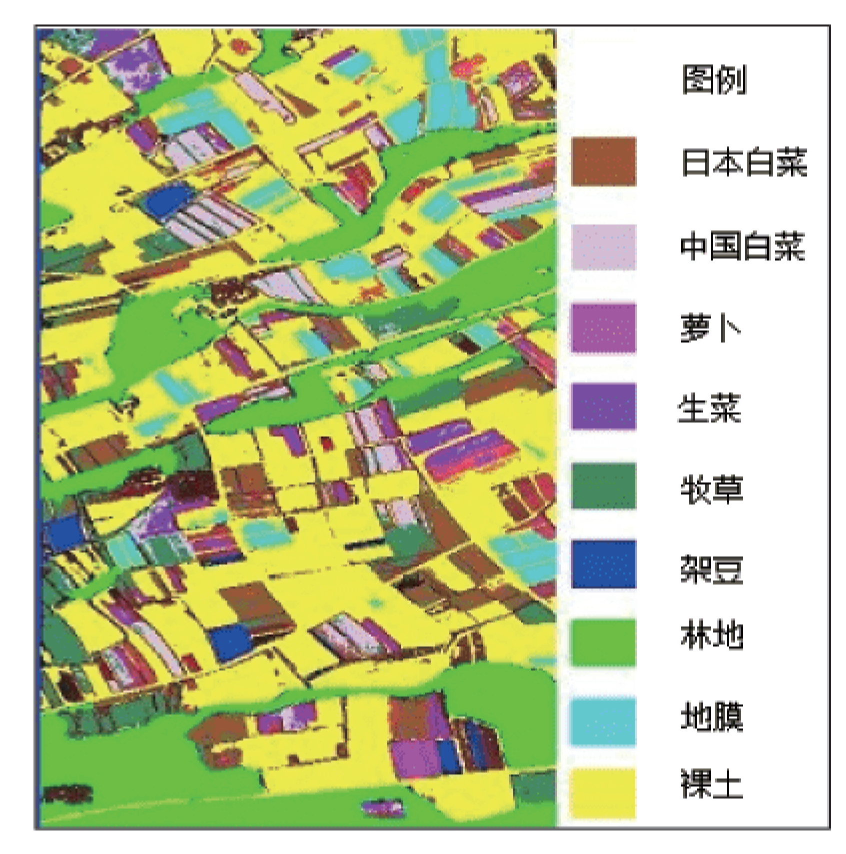
APPENDIX A

1. 图像分类

图像分类是利用计算机对图像进行定量分析，根据各自在图像信息中所反映的不同特征，把不同类别的目标区分开来的图像处理方法。图像分类示例如附图 A.1 所示。

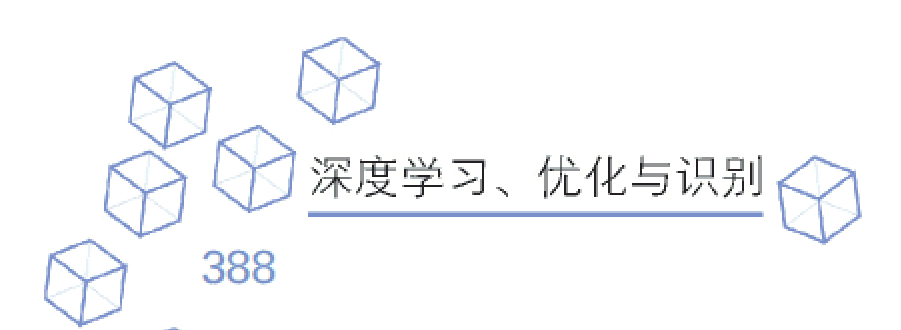


(a) 日本农田土地覆盖原始数据



(b) 日本农田土地覆盖分类结果

附图 A.1 图像分类示例

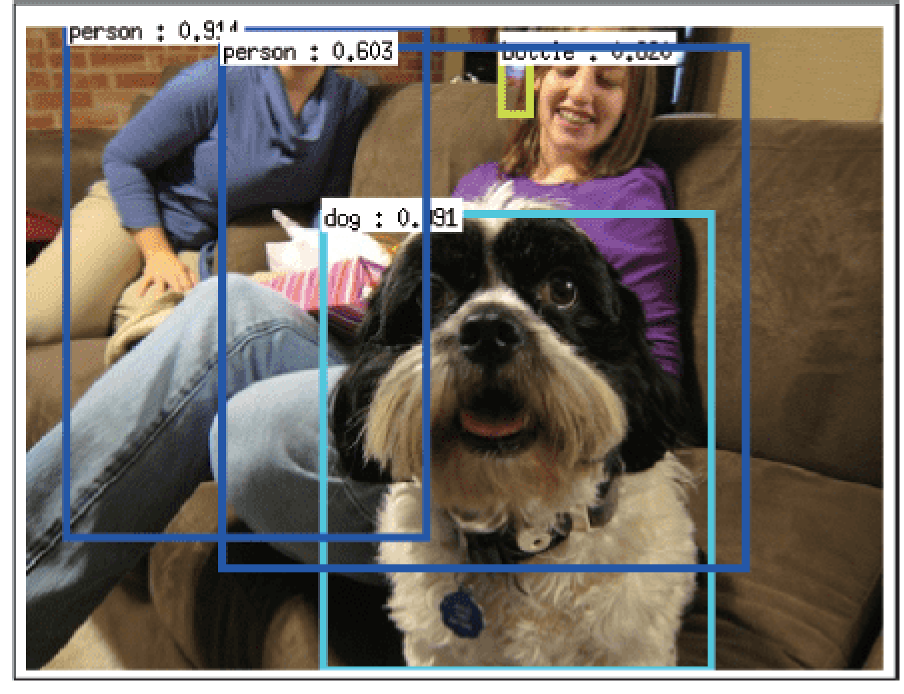


2. 目标识别

目标识别是将检测到的目标进行分类，判断属于哪一个目标类别，并在原图中标注出来，即识别出目标。目标识别示例如附图 A.2 所示。



(a) 原始图像



(b) 识别后图像

附图 A.2　目标识别示例

3. 目标检测

图像的目标检测，旨在利用图像处理与模式识别等领域的理论和方法，检测出图像中存在的目标对象，并将目标图像的位置使用边界框进行标定。目标检测示例如附图 A.3 所示。



(a) 目标检测前



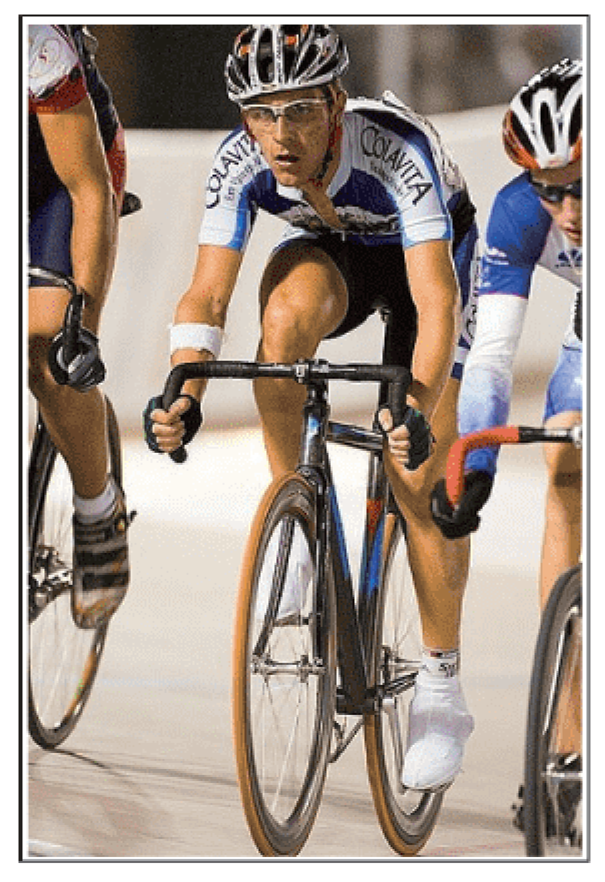
(b) 目标检测后

附图 A.3　目标检测

4. 图像分割

图像分割就是把图像分成若干个特定的、具有独特性质的区域并提出感兴趣目标的技术和过程。图像分割后提取出的目标可以用于图像语义识别、图像搜索等领域。图像分割示例如附图 A.4 所示。







(a) 原始图像

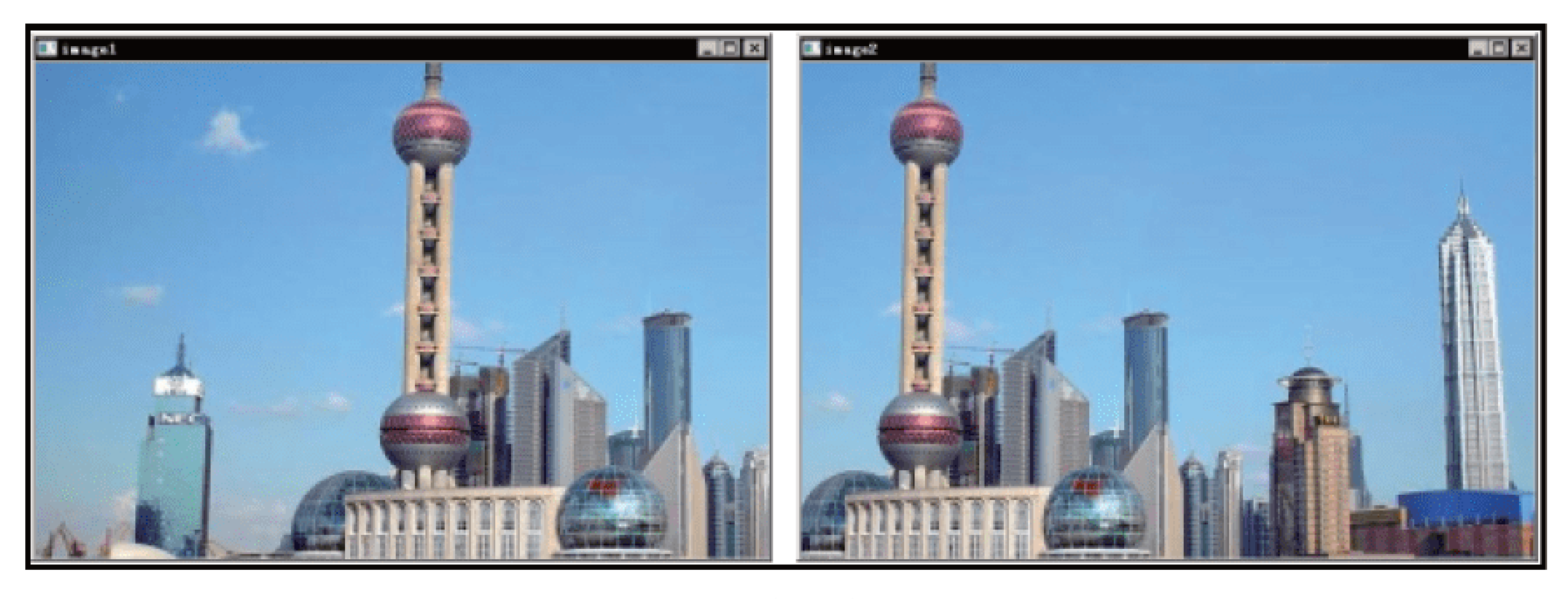


(b) 由原始图像分割生成的图像

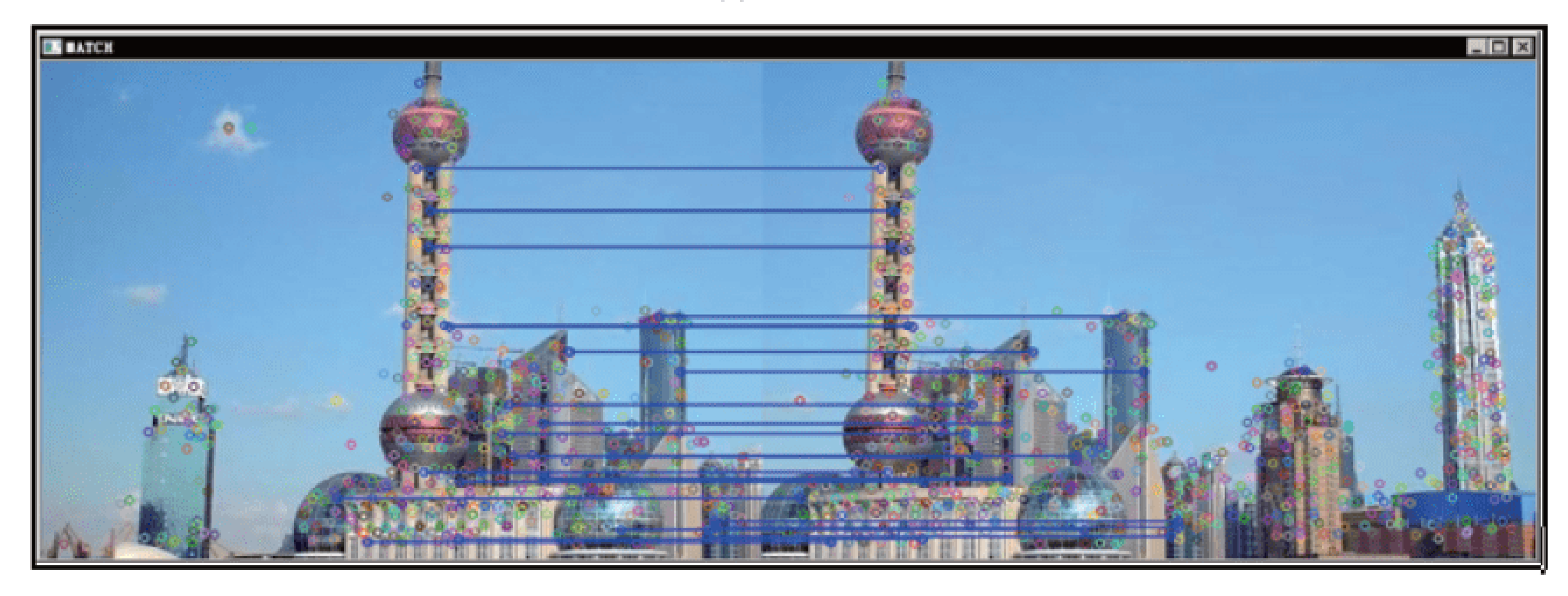
附图 A.4　图像分割示例

5. 图像配准

图像配准指的是将不同时间、不同传感器(成像设备)或不同条件下(天候、照度、摄像位置和角度等)获取的两幅或多幅图像进行匹配、叠加的过程。图像配准示例如附图 A.5 所示。



(a) 配准前

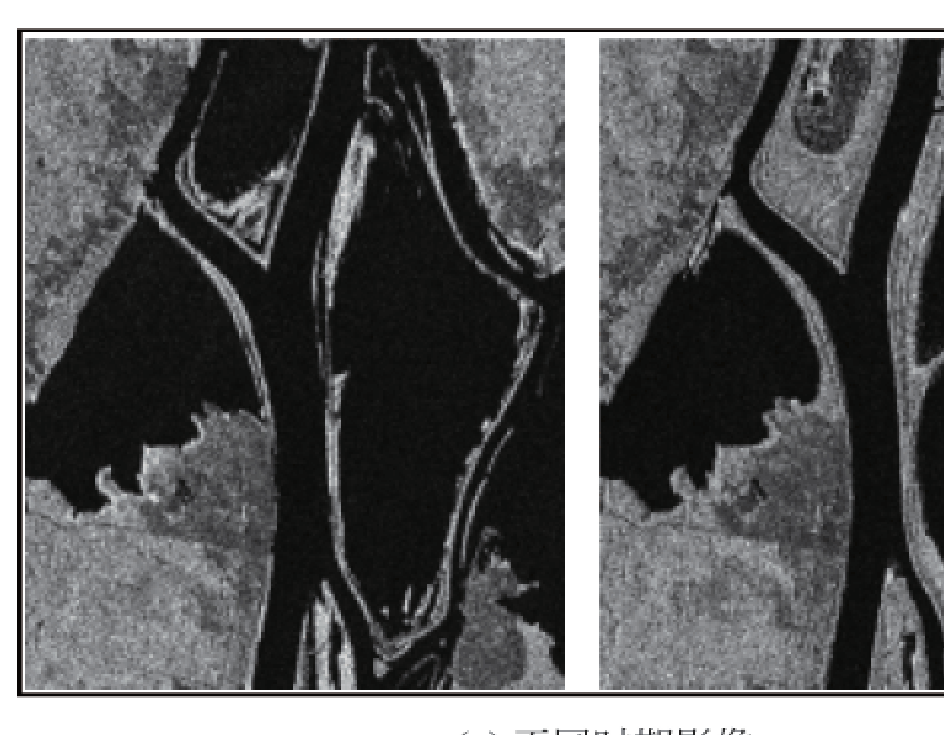


(b) 配准后

附图 A.5　图像配准示例

6. 变化检测

变化检测是指从不同时期的遥感数据中定量分析和确定地表变化的特征与过程，遥感变化检测是遥感瞬时视场中地表特征随时间发生的变化引起两个时期影像像元光谱响应的变化。变化检测示例如附图 A.6 所示。



(a) 不同时期影像



(b) 变化检测结果图

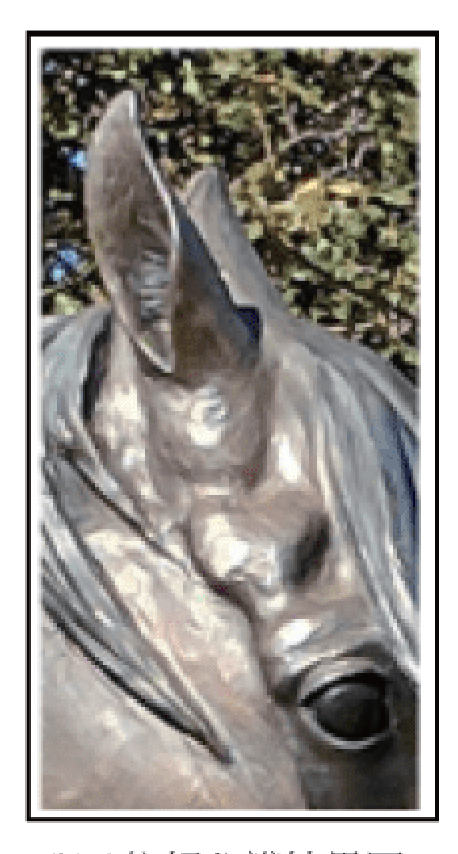
附图 A.6　变化检测示例

7. 图像超分辨

图像超分辨率任务就是给定一幅低分辨率图像或者图像序列，生成或恢复成它的高分辨率图像。按照重建的技术手段分为：基于插值的方法、基于重建的方法和基于学习的方法。图像超分辨示例如附图 A.7 所示。



(a) 原始图片

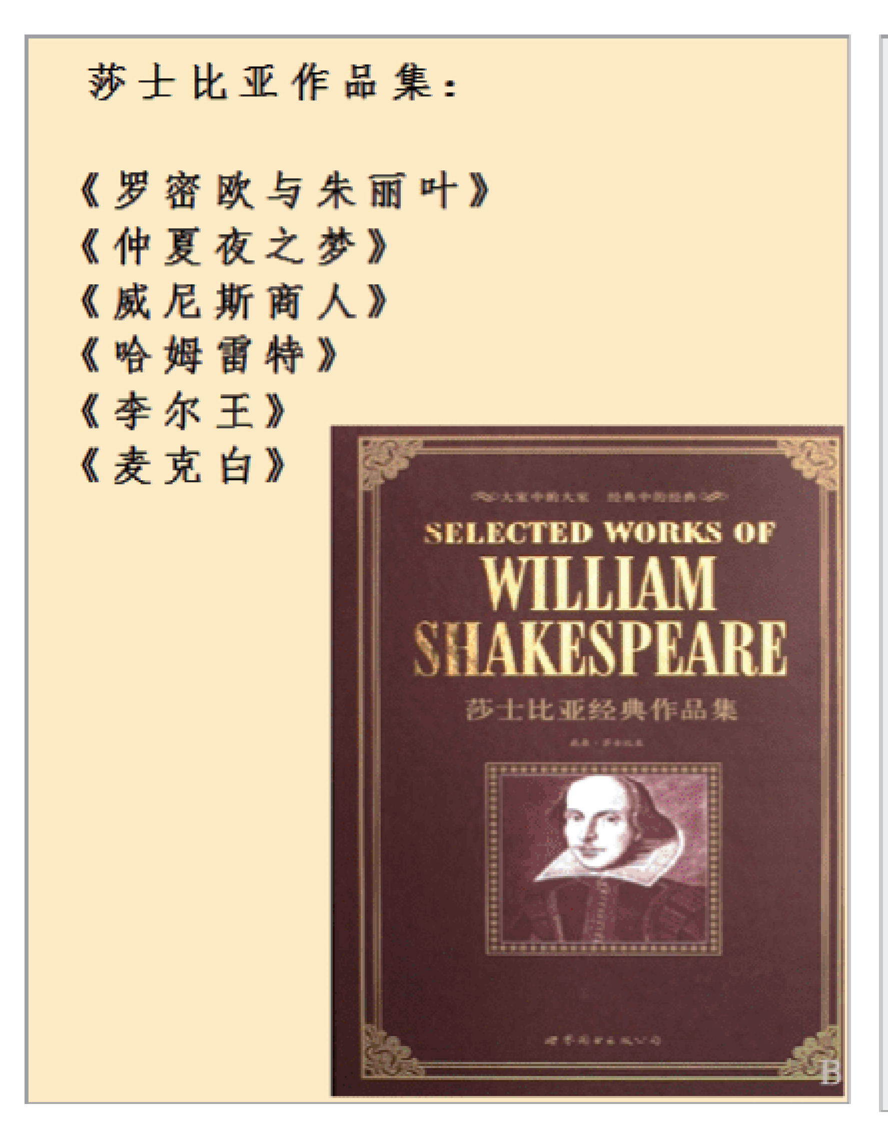


(b) 3倍超分辨结果图

附图 A.7　图像超分辨示例

8. 文本处理

文本指的是由多个词构成序列来组成句子，以及多个句子构成序列组成一个段落。文本处理具体到任务例如情感/词性分类，机器翻译或者语言生成模型等。文本处理示例如附图 A.8 所示。



(a) 文本训练数据

VIOLA:
Why, Salisbury must find his flesh and thought
That which I am not aps, not a man and in fire,
To show the reining of the raven and the wars
To grace my hand reproach within, and not a
fair are hand, That Caesar and my goodly
father's world; When I was heaven of presence
and ourfleets, We spare with hours, but cut thy
council I am great, Murdered and by thy
master's readythere My power to give thee but
so much as hell:Some service in the noble
bondman here, Would show him to her wine.
KING LEAR:
O, if you were a feeble sight, the courtesy of
your law.Your sight and several breath, will
wear the gods.With his heads, and my hands are
wonder'd at the deeds,So drop upon your
lordship's head, and your opinion Shall be
against your honour.

(b) 生成莎士比亚风格诗歌样本

附图 A.8　文本处理示例

9. 图像标注

图像标注是根据图片生成描述文字，借助“语义概念”检索感兴趣图像，在图像搜索领域有广泛应用，目前主要的实现方法有 CNN、RNN 等。图像标注示例如附图 A.9 所示。



(a) 原始图像

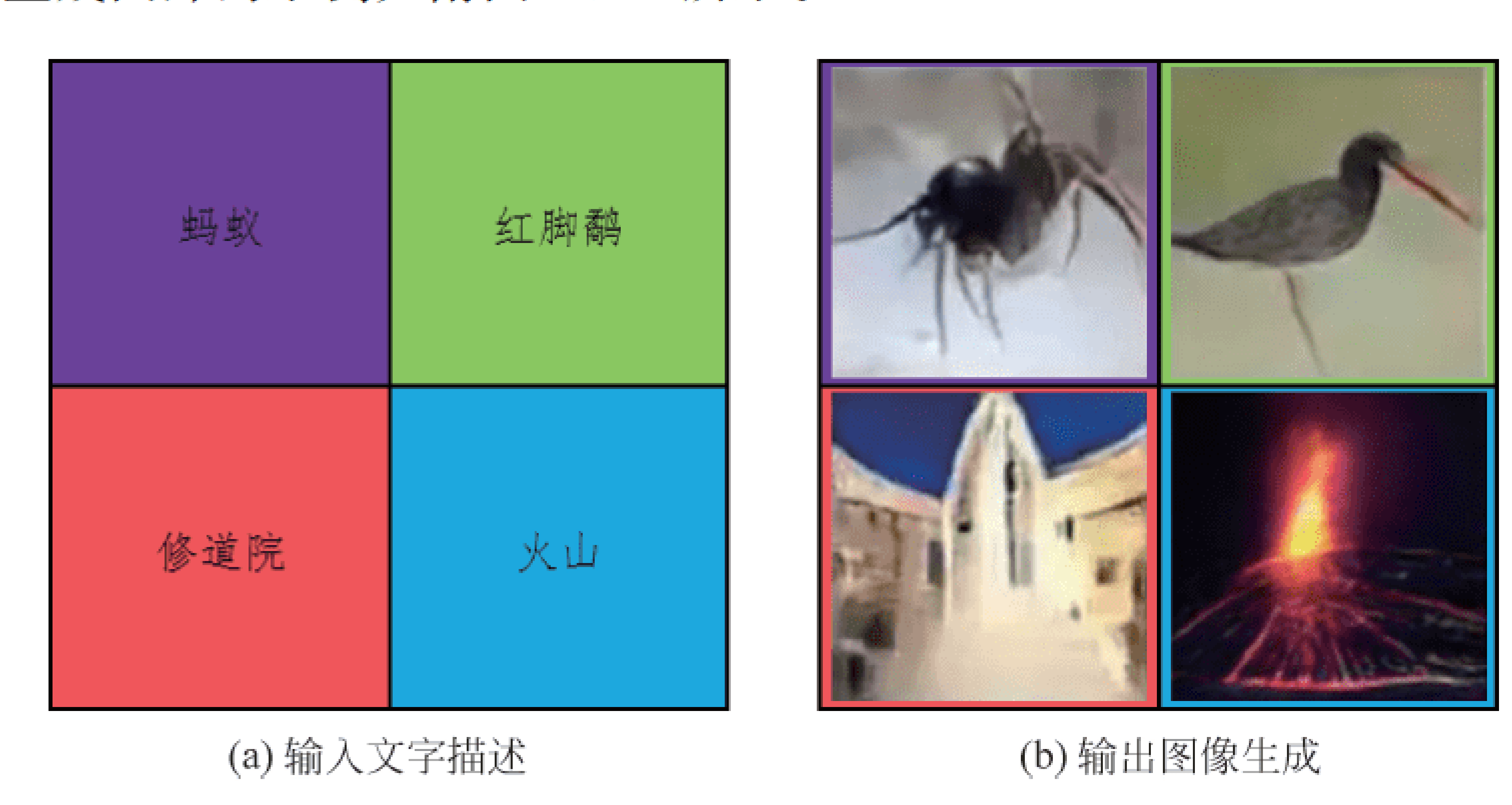


(b) 由原始图像生成文字描述

附图 A.9　图像标注示例

10. 从文字到图

既然可以实现图像标注任务，那么也能从文字生成图片. 该过程包含：①利用自然语言处理来理解输入中的描述。②利用神经网络模型输出一个准确、自然的图像，对文字进行表达。从文字生成图片的示例如附图 A.10 所示。

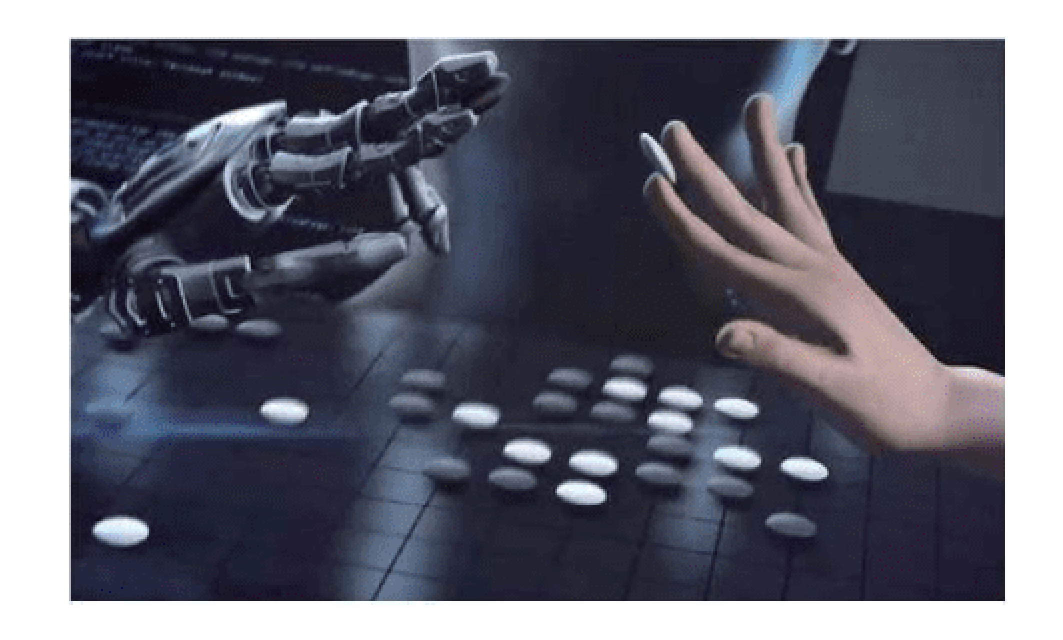


(a) 输入文字描述　(b) 输出图像生成

附图 A.10　文字生成图片示例

11. Alpha Go

Alpha Go(见附图 A.11)使用蒙特卡洛树搜索，通过自我对弈提升实力，借助价值网络与策略网络这两种深度神经网络，通过值网络来评估大量选点判断当前局势来辅助策略网络，通过策略网络企图寻找最佳的下一步并选择落点。

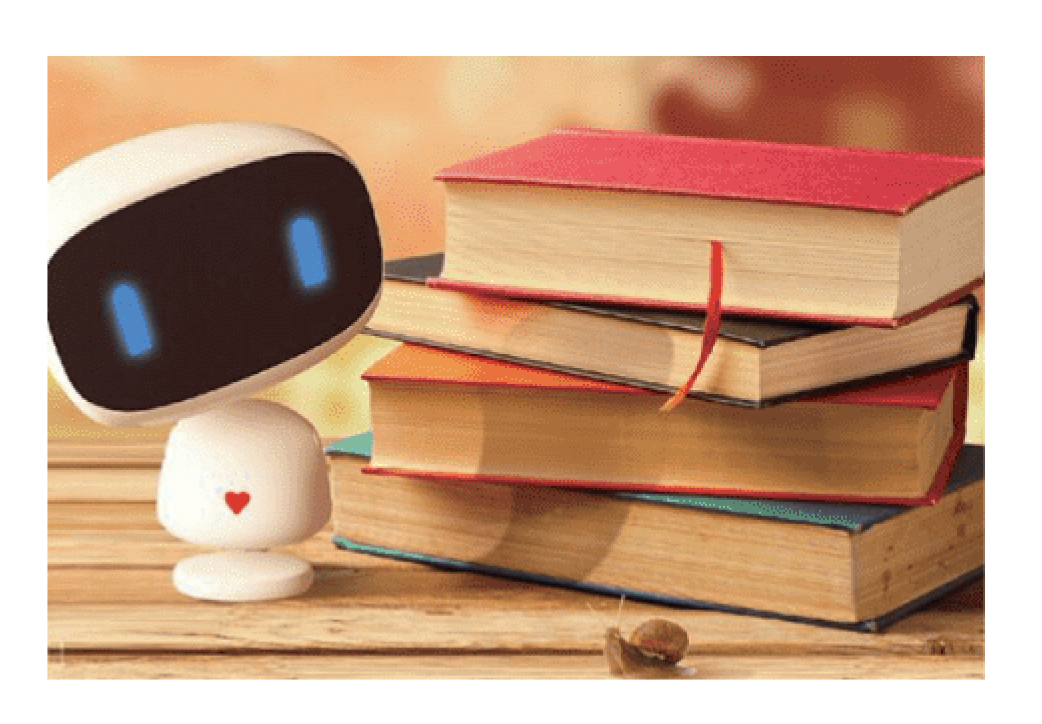


附图 A.11　人机对弈

12. 百度-小度

百度小度(见附图 A.12)利用深度学习算法，集成了自然语言理解、语音识别、机器视觉等多个领域的人工智能技术。可以通过不断地“学习”新的知识，学会更多的技能以及更加自然的人机交互，小度曾在《最强大脑》的“人机对战环节”以 0.01%的精度准确识别出了双

胞胎姐妹，从而战胜了人类选手。



附图 A.12　百度小度

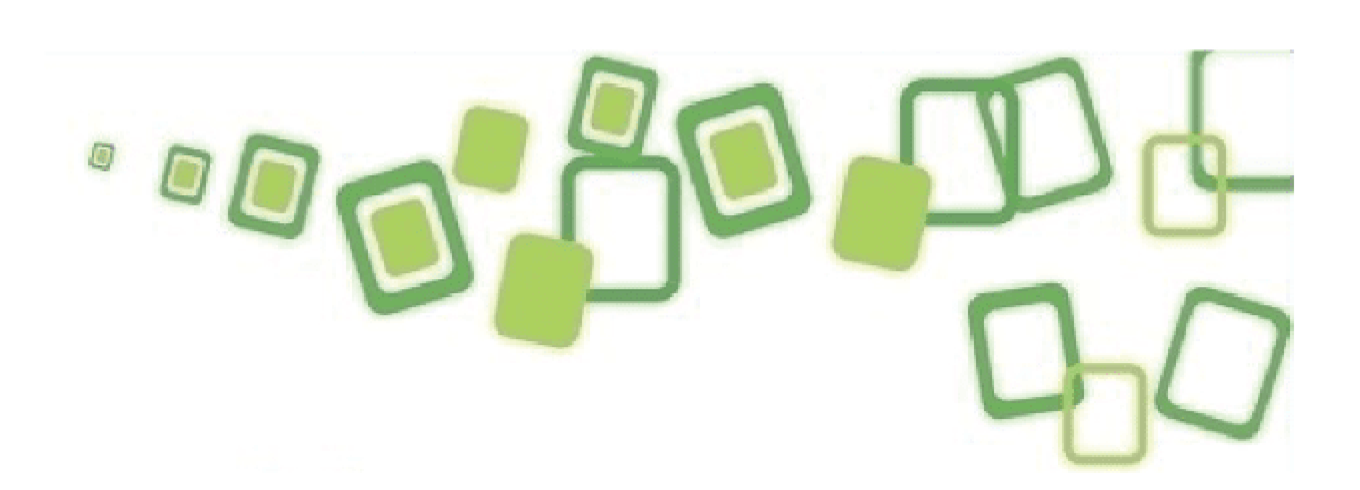
13. 无人超市执守

无人超市是由亚马逊结合人工智能做的概念性超市“Amazon Go”，如附图 A.13 所示，保留了逛的元素，不用排队，不用结账，不用注册. 结合计算机视觉，深度学习算法和传感器融合技术的“径直出门”技术，可自动甄别商品是被拿走还是被放回，并显示账单，完成自动扣款。



附图 A.13　Amazon Go 无人超市执守

附录B



代码介绍

APPENDIX B

第 2 章　深度前馈神经网络

2.1　Theano：基于 Python 的深度学习的框架和库

2.2　Javascript：基于 Javascript 的深度学习的框架和库

2.3　Keras：基于 Python 的深度学习的框架和库

2.4　Ruby：基于 Ruby 的深度学习的框架和库

2.5　Objective-c：基于 Objective-c 的深度学习的框架和库

2.6　DeepLearnToolbox-master：基于 Matlab 的深度学习的框架和库

第 3 章　深度卷积神经网络

3.1　CNN：卷积神经网络

3.2　RCNN and SPPnet：区域卷积神经网络和空间金字塔池化网络

3.3　Fast rcnn：快速区域卷积神经网络

3.4　Alexnet：经典的卷积神经网络

3.5　FCN：全卷积网络

第 4 章　深度堆栈自编码网络

4.1　CDBN：卷积深度置信网络

4.2　dbn_tf：基于 TensorFlow 的深度玻尔兹曼机网络



4.3　deep_boltzmann：深度玻尔兹曼机网络
4.4　Sparse-Autoencoder-Tensorflow：基于 TensorFlow 的稀疏自编码网络
4.5　StackedDAE：堆栈降噪自编码网络
4.6　TensorFlow-DeepAutoencoder：基于 TensorFlow 的深度自编码网络
4.7　TensorFlow-VAE：基于 TensorFlow 的变分自编码网络

第 5 章　稀疏深度神经网络

5.1　sparseae_exercise：基于 Matlab 的稀疏自编码网络

第 6 章　深度融合网络

6.1　Python-ELM：基于 Python 的极限学习机以及深度极限学习机

第 7 章　深度生成网络

7.1　WGAN-tensorflow：带有 Wasserstein 距离的生成式对抗网络
7.2　LS-GAN：最小二乘生成式对抗网络
7.3　DCGAN-tensorflow：深度卷积生成式对抗网络

第 8 章　复卷积神经网络与二值神经网络

8.1　Deep CCNN：深度复卷积网络
8.2　BinaryNet：深度二值神经网络

第 9 章　深度循环和递归神经网络

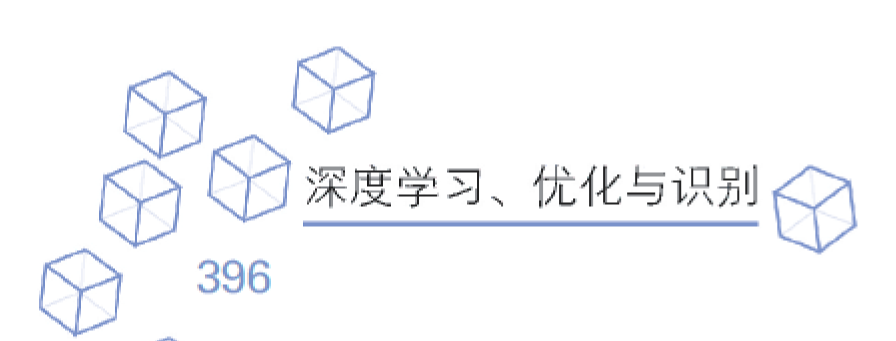
9.1　char-rnn：深度循环和递归神经网络

第 10 章　深度强化学习

10.1　deer：强化学习的框架和库
10.2　deep-q-learning：深度 Q 学习

第 11 章　深度学习软件仿真平台及开发环境

11.1　Caffe-Alexnet：基于 Caffe 的 Alexnet 网络仿真实现
11.2　MXNet-VGG：基于 MXNet 的 VGG 网络仿真实现
11.3　Tensorflow-GAN：基于 Tensorflow 的生成式对抗网络仿真实现
11.4　theano-lstm：基于 theano 的长短时记忆网络仿真实现
11.5　Torch7-binarynet：基于 Torch7 的二值神经网络网络仿真实现



第 12 章　基于深度神经网络的 SAR 和 PolSAR 影像地物分类

12.1　GAN_PolSAR：基于生成式对抗网络的 PolSAR 影像地物分类
12.2　Deep CCNN_PolSAR：基于深度复数卷积网络的 PolSAR 影像地物分类
12.3　Deep Residual Network_PolSAR：基于深度残差网络的 PolSAR 影像地物分类

第 14 章　基于深度神经网络的高光谱图像分类与压缩

14.1　deeplearn_HSI：基于深度学习的高光谱图像分类

第 15 章　基于深度学习的目标检测与识别

15.1　yolo：you only look once 目标检测算法源代码
15.2　voc-dpm-release5：dpm 目标检测算法源代码
15.3　rcnn：基于区域卷积神经网络的目标检测算法源代码
15.4　fast-rcnn：基于快速区域卷积神经网络的目标检测算法源代码
15.5　faster_rcnn：基于更快区域卷积神经网络的目标检测算法源代码
15.6　caffe：caffe 深度学习工具包
15.7　caffe-fast-rcnn：基于快速区域卷积神经网络的 caffe 源代码
15.8　caffe-ssd：ssd 目标检测算法的 caffe 源代码